1 Objectif

« Text mining » avec Knime et RapidMiner. Classement automatique de nouvelles.

L'approche statistique du « text mining » consiste à transformer une collection de documents textuels en une matrice de valeurs numériques sur laquelle nous pouvons appliquer les techniques d'analyse de données. Bien évidemment, d'autres prismes existent. Je préfère prendre mes précautions avant la levée de bouclier des linguistes et autres tenants des approches sémantiques. Il y a de la place pour tout le monde.

Le terme « document non-structuré » est souvent évoqué lorsque l'on parle de texte. Cela ne signifie pas qu'il n'obéit pas à une certaine organisation (titres, chapitres, paragraphes, questions-réponses, etc.). Il indique avant tout que l'on ne peut pas exprimer directement la collection sous la forme d'un tableau individus – variables que l'on manipule usuellement en data mining. Pour parvenir à cette représentation, il est nécessaire de passer par une phase de prétraitement qui implique, avant la phase de modélisation proprement dite, des choix et des partis pris qui influenceront la qualité des résultats.

Dans ce tutoriel, je reprends un des exercices de catégorisation de textes (fouille de textes) que j'encadre en Master IDS-SISE[1] du Département Informatique et Statistique de l'Université Lumière Lyon 2. Nous effectuons la totalité des opérations sous R. L'utilisation des packages spécialisés 'XML' et 'tm' facilitent grandement les opérations, avouons-le. Je me suis demandé s'il était possible de réaliser les mêmes traitements à l'aide d'autres logiciels libres. J'ai beaucoup cherché. Trouver de la documentation qui corresponde véritablement à ce que je souhaitais mettre en place n'a pas été facile (et encore, je savais exactement ce qu'il y avait à faire, ça aide pour les recherche sur le web). J'ai finalement réussi à reproduire (à peu près) la totalité de la séance sous les logiciels **Knime 2.9.1** et **RapidMiner 5.3**.

Ces logiciels sont bien connus du monde libre[2] (RapidMiner l'est de moins en moins puisque la version gratuite STARTER est volontairement bridée). Ils sont néanmoins relativement peu utilisés en France, notamment parce que leurs sorties ne correspondent pas toujours aux standards du domaine. Nous verrons dans ce tutoriel qu'ils disposent de bibliothèques spécialisées pour l'appréhension du texte à des fins d'analyse statistique. Le tout est de pouvoir identifier les outils et les paramétrages adéquats.

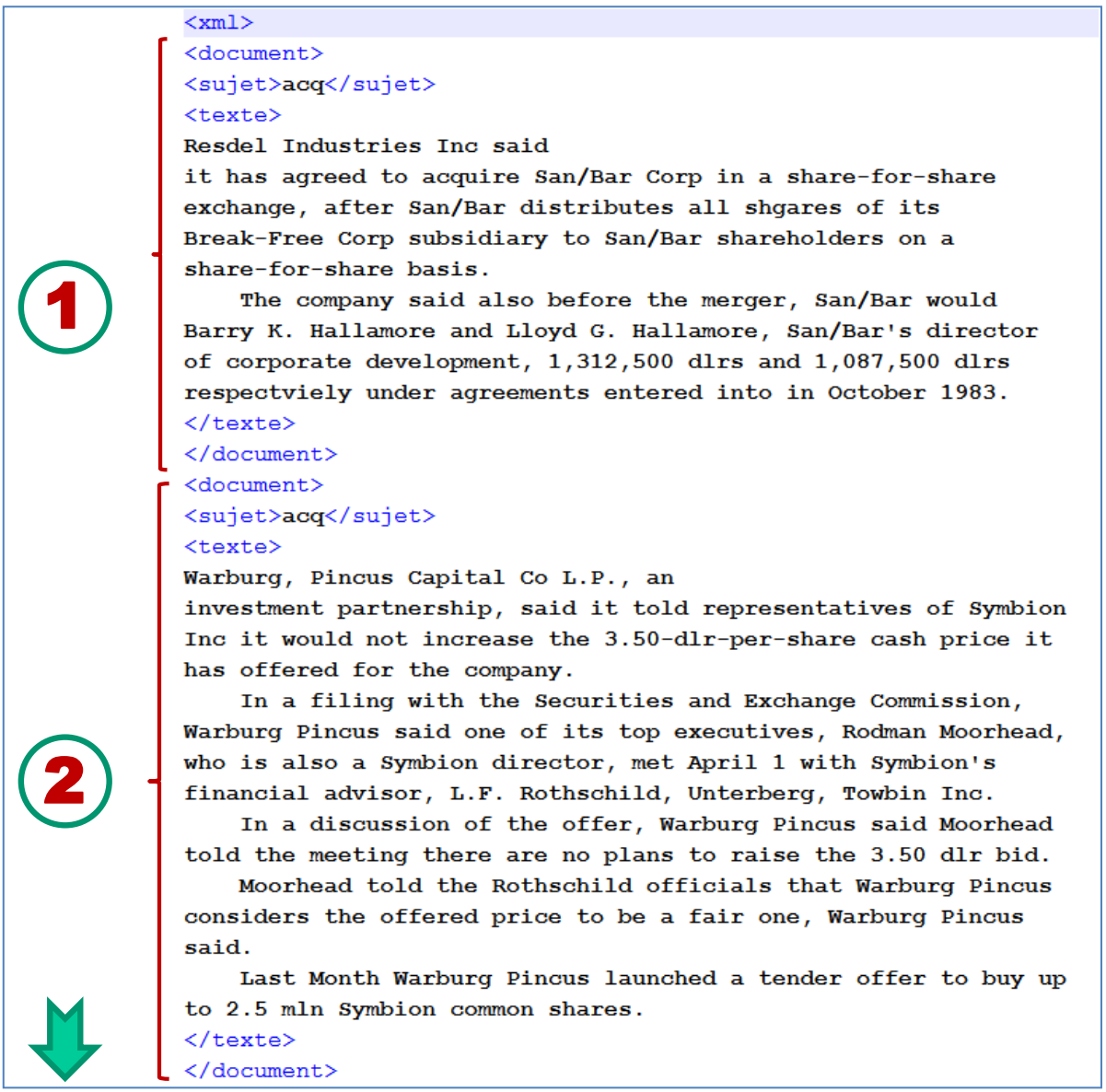
[1] http://dis.univ-lyon2.fr/?page_id=195

[2] http://www.kdnuggets.com/polls/2013/analytics-big-data-mining-data-science-software.html

2 Données – Les nouvelles de Reuters

Nous utilisons la base Reuters[3], un benchmark bien connu de la « **catégorisation de textes** ». Le fichier « **reuters.xml** » au format XML a été nettoyé de manière à simplifier les traitements. Il est composé de 117 documents. A chaque document est associé un sujet (2 thèmes possibles : {acq, crude}) et un texte. L'objectif est de construire une fonction de prédiction permettant d'assigner automatiquement les sujets aux textes. Nous nous situons dans un schéma d'apprentissage supervisé où la variable cible SUJET est binaire.

Voici les 2 premières observations de notre collection :



Nous remarquons la structure hiérarchique du fichier XML :

[3] http://archive.ics.uci.edu/ml/datasets/Reuters-21578+Text+Categorization+Collection

```
<xml>
	<document>
		<sujet>
		...
		</sujet>
		<texte>
		...
		</texte>
	</document>
	...
</xml>
```

La manipulation de ce fichier implique plusieurs enjeux :

1. Il faut le parser et le charger dans une structure adéquate en dissociant le sujet et le texte pour chaque enregistrement (document).
2. Les sujets peuvent être collectés tels quels dans un vecteur, il n'y a pas de difficultés particulières. Il est dès lors possible d'effectuer des calculs statiques simples. On observe ainsi que 71 (resp. 46) documents correspondent au sujet « acq » (resp. « crude »).
3. Les textes également peuvent être collectés dans des vecteurs. Mais il n'est pas possible d'effectuer de traitements statistiques à ce stade.
4. Pour qu'ils soient réalisables, nous devons transformer chaque texte en un vecteur où les éléments sont étiquetés par des « termes », communs à tous les documents, auxquels sont attribués des valeurs que l'on appelle « pondérations », spécifiques au document traité.
5. Il reste alors à consolider toutes ces informations dans une matrice de données où, pour chaque ligne (document), nous observons sa classe (sujet) et les pondérations associées aux termes (colonnes). On parle alors de « document term matrix »[4].

Dans la copie d'écran ci-dessous, nous montrons les premières lignes et colonnes d'un tableau – extrait sous R à l'aide du package 'tm', les résultats seront un peu différents dans Knime et RapidMiner selon les prétraitements réalisés – comportant n = 117 lignes (parce qu'il y a 117 documents), p = 2315 termes (p + 1 = 2316 colonnes en incluant le sujet), avec la pondération TF (term frequency) qui consiste simplement à compter le nombre d'occurrence de chaque terme dans les documents[5].

[4] http://en.wikipedia.org/wiki/Document-term_matrix

[5] Voir « Matrice des occurrences », http://fr.wikipedia.org/wiki/Analyse_sémantique_latente (notre tableau est transposé par rapport à ce qui est décrit dans la référence).

sujet	sanbar	corp	dlrs	hallamor	shareforshar	acquir	agre	agreement
acq	5	2	2	2	2	1	1	1
acq	0	0	0	0	0	0	0	0
acq	0	2	7	0	0	1	0	0
acq	0	1	0	0	0	0	0	0
acq	0	0	1	0	0	2	0	0

Par exemple, le terme « **sanbar** » est apparu 5 fois dans le premier document, « **corp** » 2 fois, « **dlrs** » 2 fois également, etc. Voyons ce qu'il en est réellement lorsque nous revenons sur le document original.

```
<document>
<sujet>acq</sujet>
<texte>
Resdel Industries Inc said
it has agreed to acquire San/Bar Corp in a share-for-share
exchange, after San/Bar distributes all shgares of its
Break-Free Corp subsidiary to San/Bar shareholders on a
share-for-share basis.
    The company said also before the merger, San/Bar would
Barry K. Hallamore and Lloyd G. Hallamore, San/Bar's director
of corporate development, 1,312,500 dlrs and 1,087,500 dlrs
respectviely under agreements entered into in October 1983.
</texte>
</document>
```

Nous observons qu'il y a une correspondance approximative entre le terme « sanbar », utilisé dans le tableau de données, et le mot « San/Bar » observé dans le texte brut. Cela résulte du nettoyage effectué (transformation de la casse, stemming[6], lemmatisation[7], retrait des stopwords[8], etc.) avant la génération du tableau final. L'objectif est de réduire le nombre de termes (nombre de colonnes) de la matrice afin de réunir des indications plus compactes, plus pertinentes, sur l'information véhiculée par les documents.

Les choix des « termes » et du type de « pondération » sont déterminants pour la qualité de la modélisation à venir.

3 Catégorisation de textes avec Knime

Nous utilisons la version Knime Desktop 2.9.1 accessible gratuitement sur le site de l'éditeur (http://www.knime.org/knime). Nous détaillons le processus aboutissant à la génération de la matrice « documents-termes » (matrice DT) qui sera utilisée pour la construction d'un

[6] http://fr.wikipedia.org/wiki/Racinisation

[7] http://fr.wikipedia.org/wiki/Lemmatisation

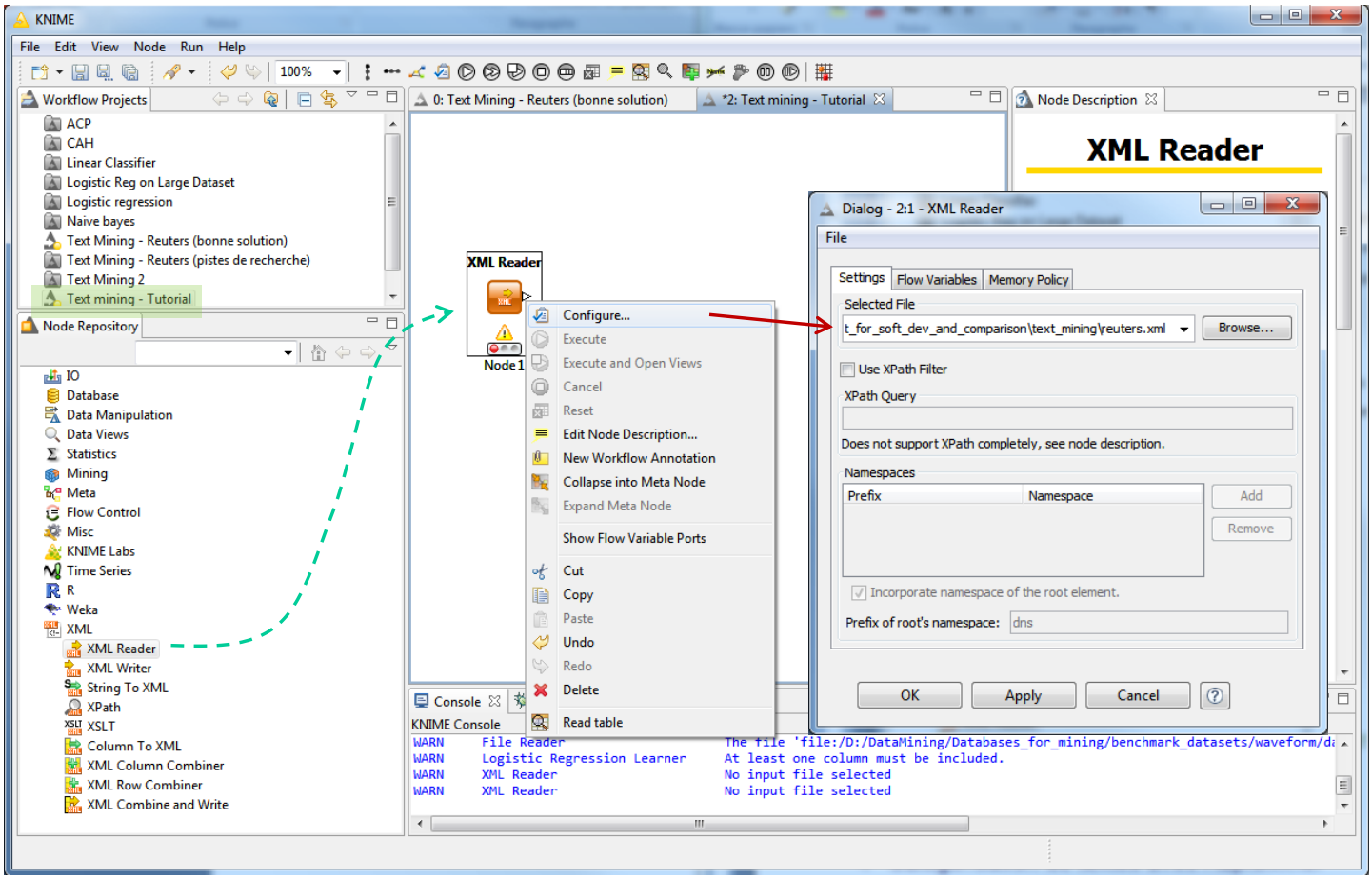
[8] http://fr.wikipedia.org/wiki/Mot_vide

classifieur associant les textes aux sujets. En voici les principales étapes : lecture et parsage[9] du fichier XML, préparation des documents de manière à réduire le nombre de termes à extraire, extraction des termes, choix du type de pondération et, enfin, élaboration de la matrice documents-termes.

Note : Je ne doute pas qu'il existe une manière plus directe de mener ces traitements. Mais, étant dans j'ai l'obligation d'expliquer chaque opération, je dois rester le plus schématique (scolaire) possible pour que tout un chacun puisse retracer et comprendre les étapes.

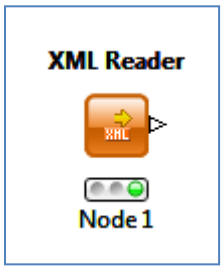
3.1 Importation du fichier XML

Après avoir démarré KNIME, un nouveau 'workflow' est créé (menu FILE / NEW). Nous le nommons « Text Mining – Tutorial ». Nous insérons le composant XML / XML Reader dans l'espace de travail. Nous le paramétrons (menu contextuel « Configure ») de manière à charger le fichier « reuters.xml ».

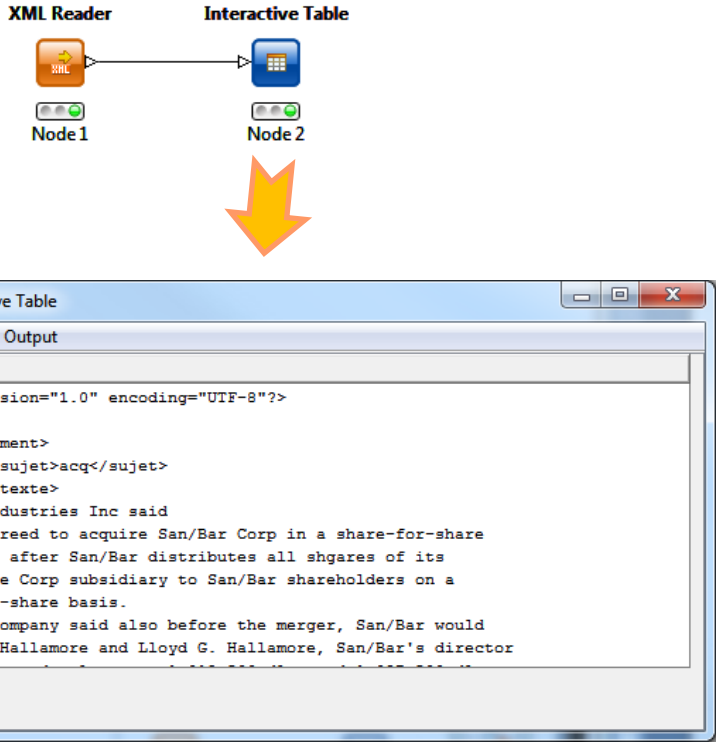


[9] Si, si, ça se dit... http://fr.wiktionary.org/wiki/parsage

Nous actionnons le menu contextuel « Execute ». Le témoin lumineux passe au vert si l'opération est couronnée de succès.

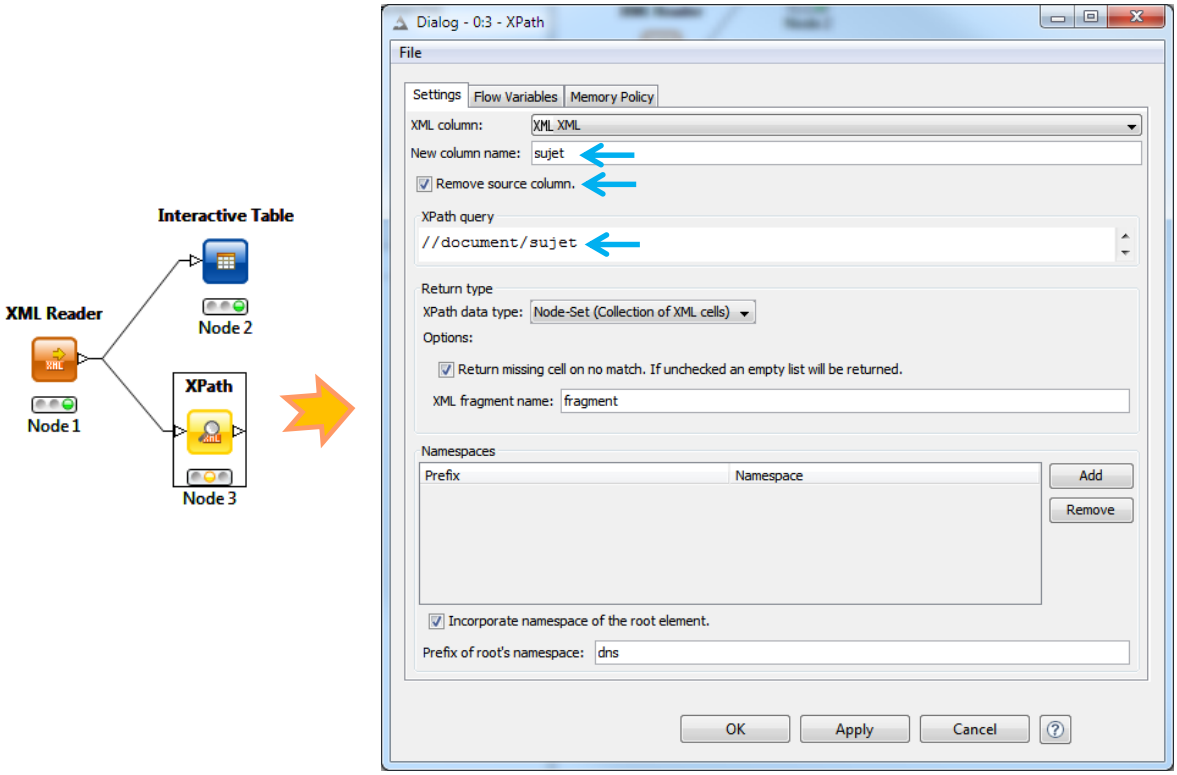


Nous utilisons le composant DATA VIEWS / INTERACTIVE TABLE pour visualiser le contenu téléchargé. Nous actionnons le menu contextuel « Execute and Open Views ». Toutes les données sont stockées dans un vecteur de taille 1. La décomposition par document n'est pas opérante pour l'instant.

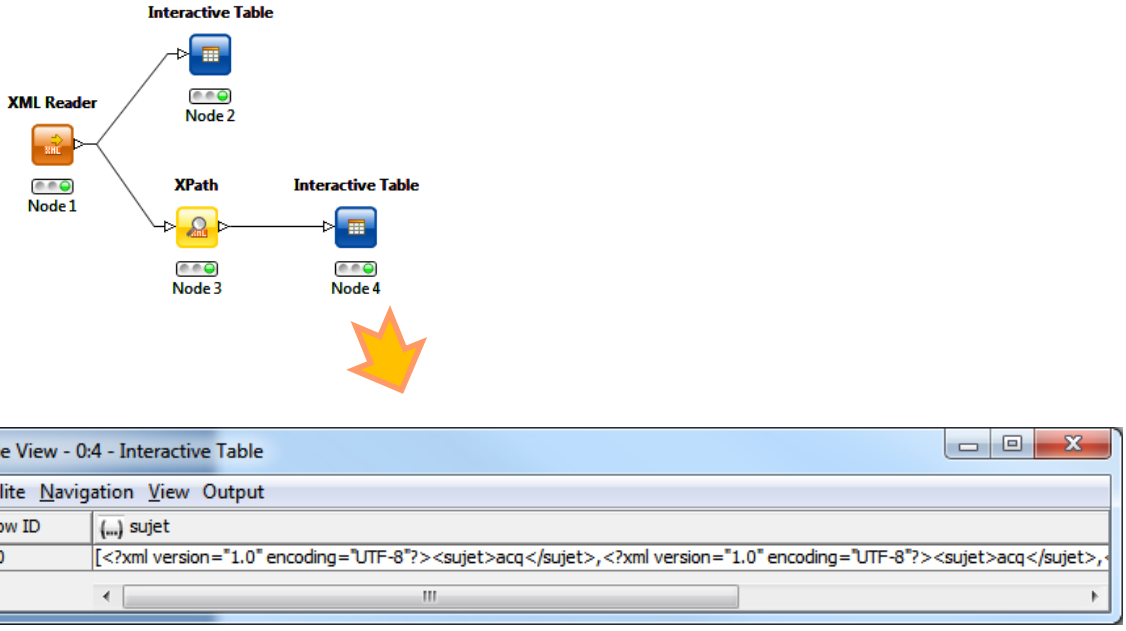


3.2 Extraction du vecteur des sujets

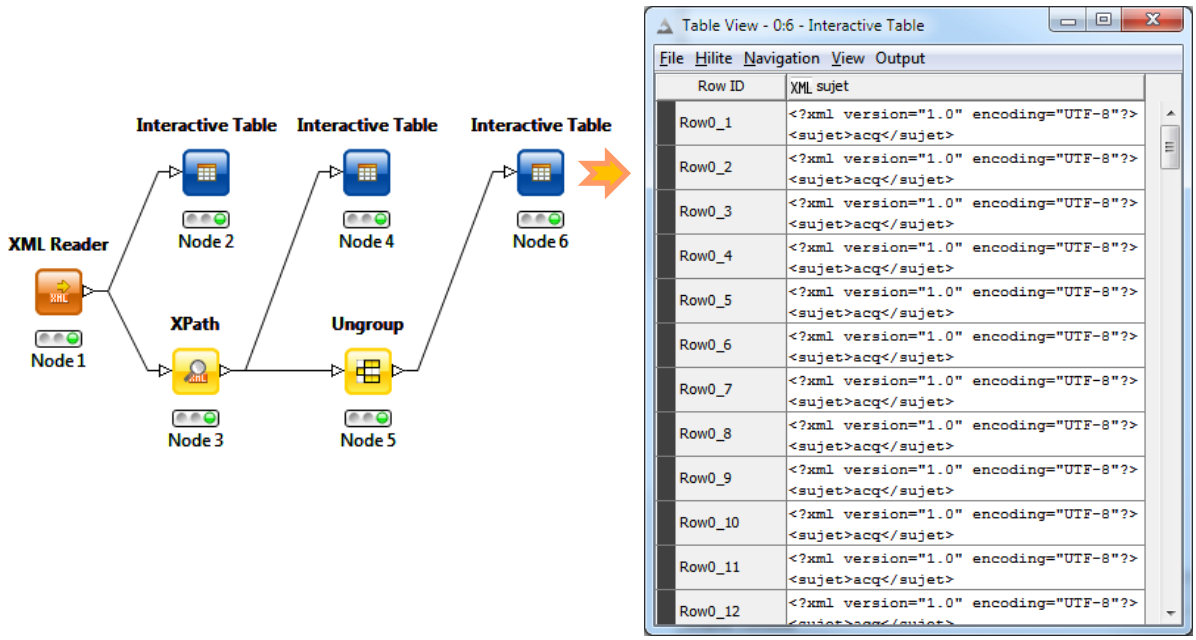
Nous utilisons l'outil XML / XPATH pour extraire les sujets associés aux documents. Le paramétrage (menu contextuel **Configure**) est très important ici : dans « Xpath Query », nous précisons le champ à extraire ; le vecteur est nommé « sujet » (New column name) ; il remplace les données sources (Remove Source Column).



Un nouveau composant « Interactive Table » nous permet de vérifier le bon fonctionnement de l'opération. Une seule ligne contenant tous les sujets est disponible pour l'instant.

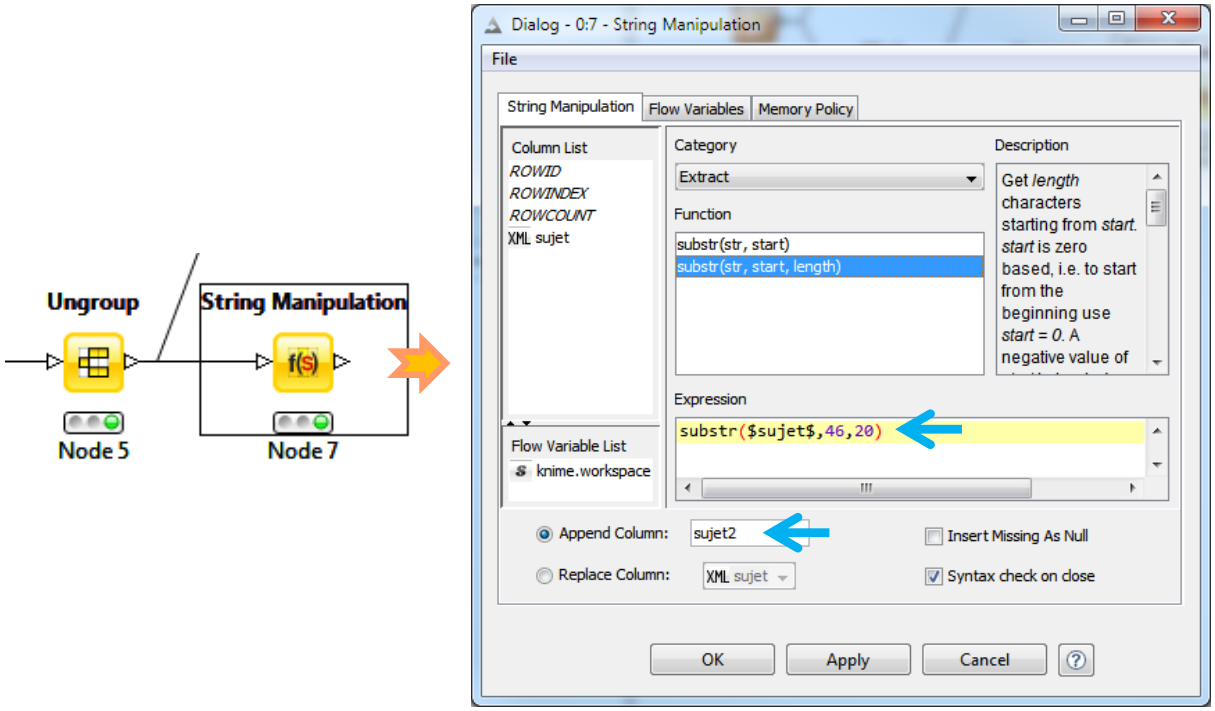


Le rôle du composant DATA MANIPULATION / ROW / TRANSFORM / UNGROUP (il n'y a pas de paramétrage particulier à faire) est de dispatcher les sujets dans un vecteur à 117 valeurs. Visualisons le résultat.

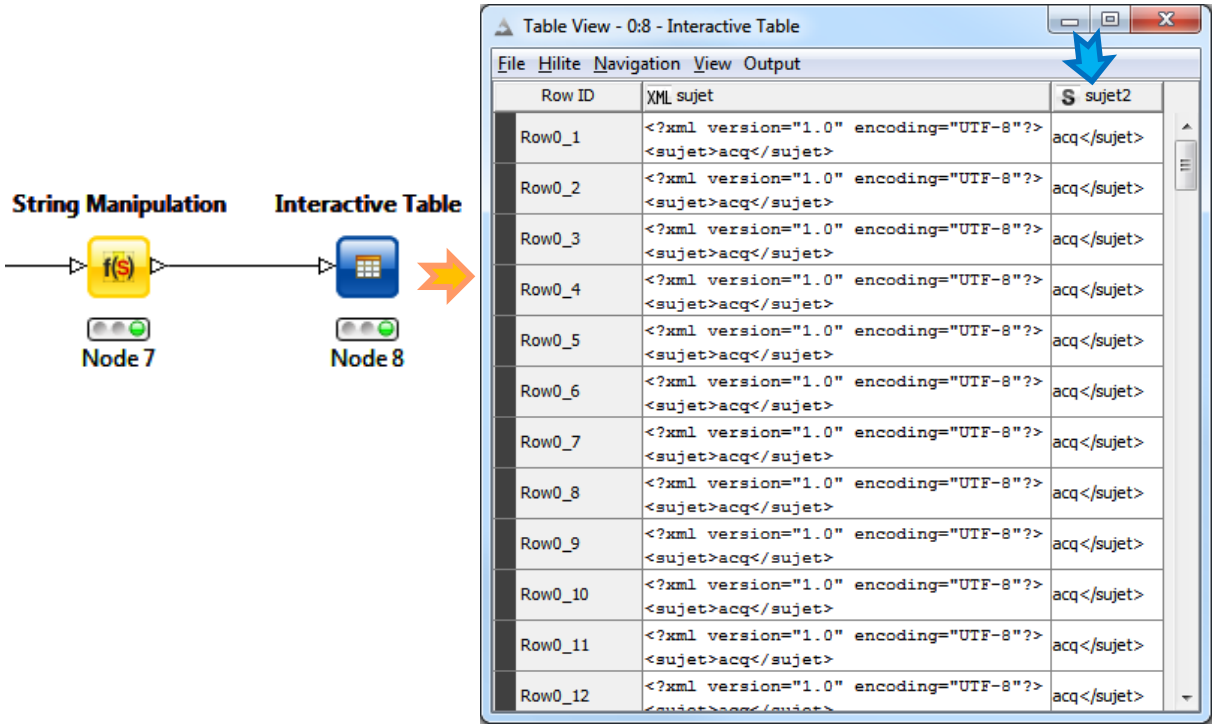


Il y a bien 117 lignes dans le tableau. On constate en revanche que les cellules sont parsemées d'informations superflues pour l'analyse. Il y a un nettoyage à faire.

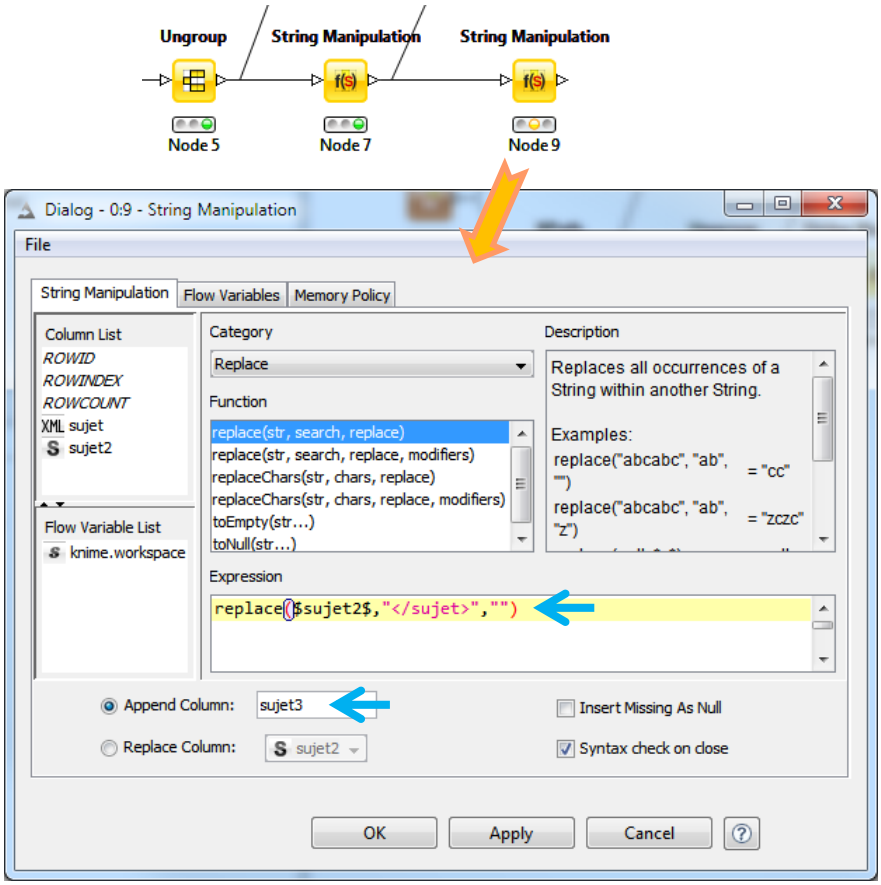
Nous procédons en deux temps. Tout d'abord, nous isolons la partie située après **<sujet>** de la chaîne de caractères à l'aide du composant DATA MANIPULATION / COLUMN / TRANSFORM / STRING MANIPULATION. Il faut considérer avec attention le paramétrage ici. La nouvelle colonne est appelée « sujet2 ».



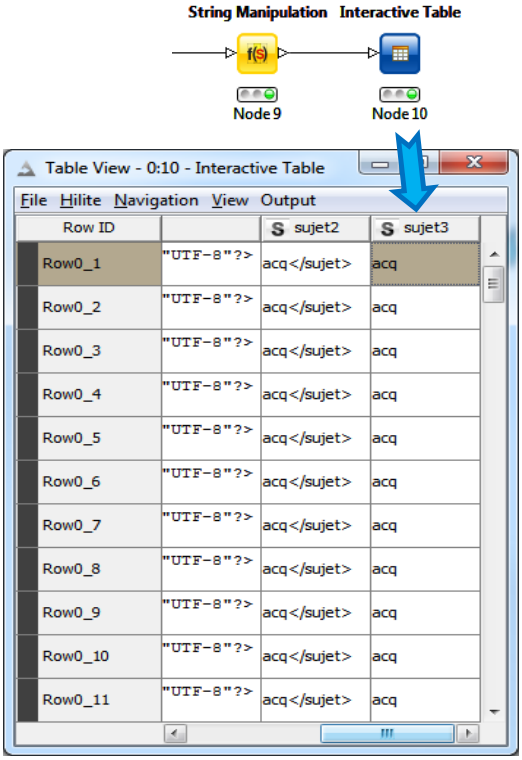
La colonne « sujet2 » se présente comme suit maintenant :



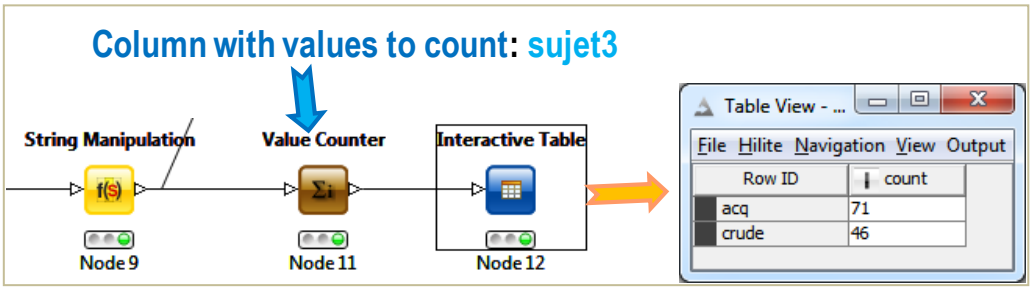
Il faut ensuite, avec un second composant STRING MANIPULATION, supprimer la partie **</sujet>** en la remplaçant par une chaîne vide.



Voyons ce qu'il en résulte.

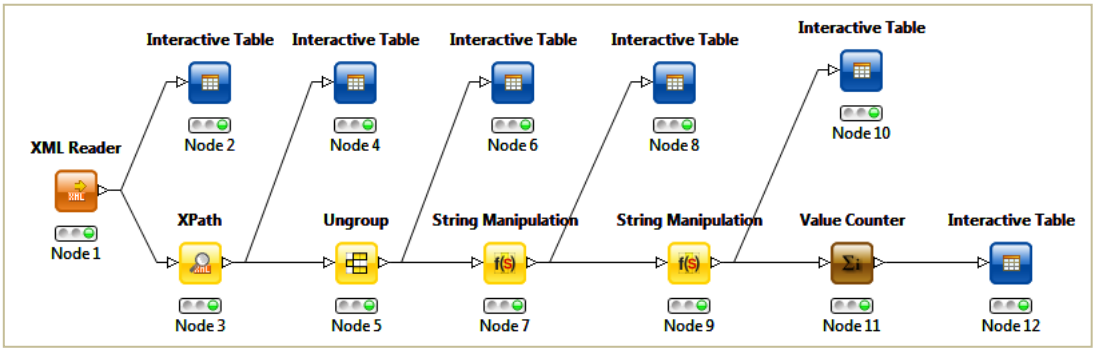


Nous pouvons effectuer un premier traitement. Nous calculons la distribution de fréquences des sujets à l'aide de l'outil STATISTICS / VALUE COUNTER. Nous sélectionnons la colonne « sujet3 » lors du paramétrage.



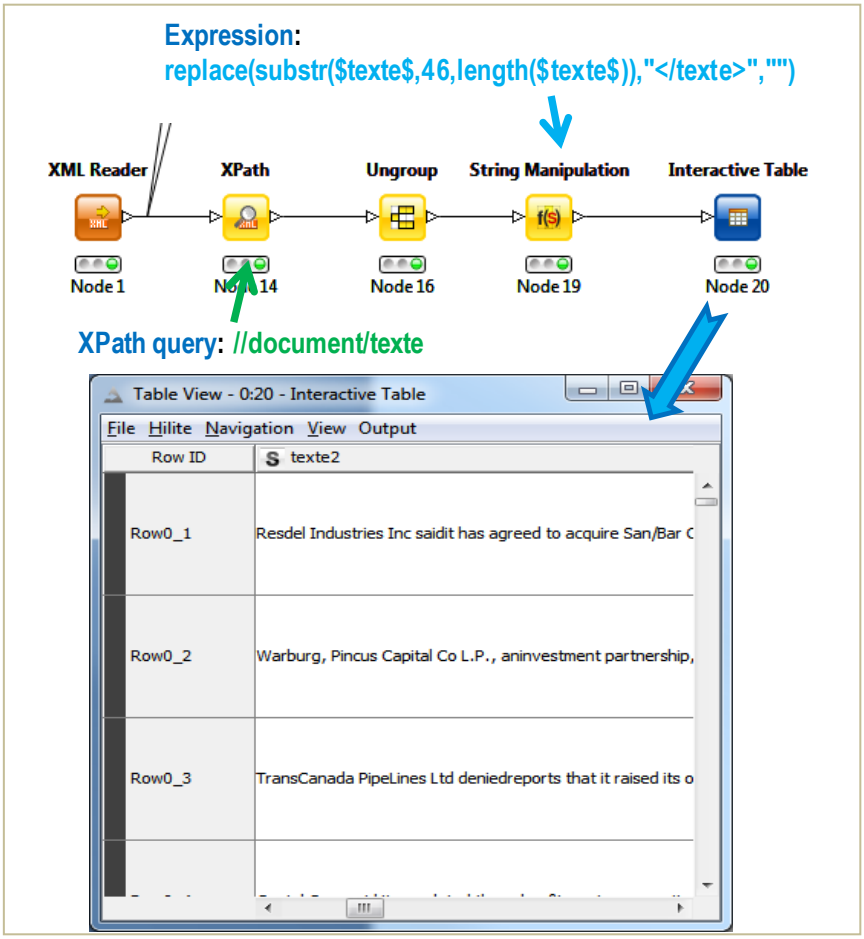
71 (resp. 46) nouvelles ont pour thème « acq » (resp. « crude »).

De nombreux composants ont été placés dans l'espace de travail, voici la chaîne de traitements à ce stade de notre analyse.



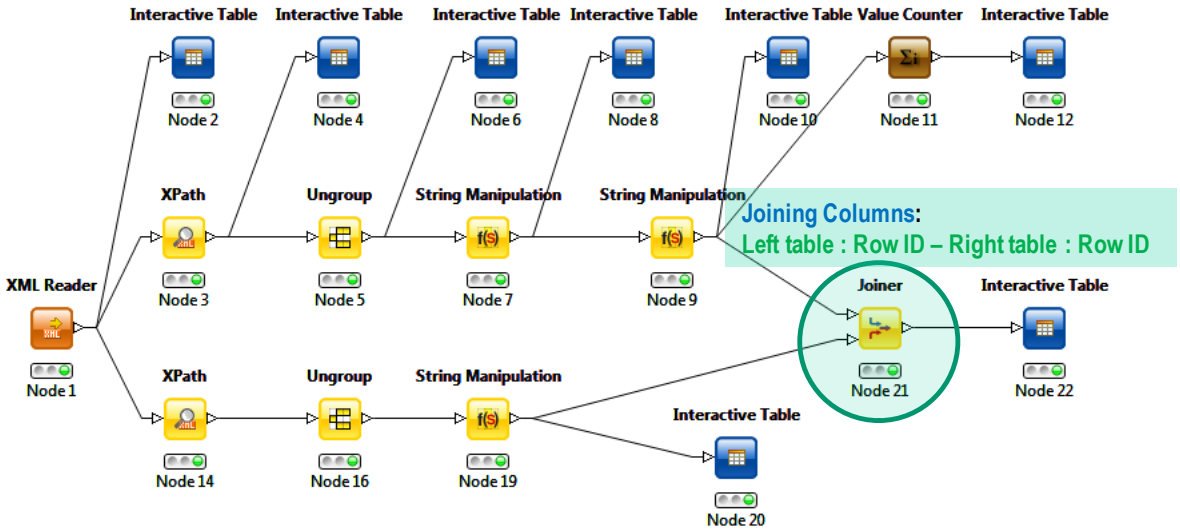
3.3 Extraction du vecteur des textes

En suivant la même démarche, nous allons extraire la partie située entre les balises <texte> et </texte> et les stocker dans un second vecteur. Nous construisons la séquence suivante. Le point de départ est le « XML Reader » accédant au fichier « reuters.xml ».



3.4 Jointure des deux vecteurs

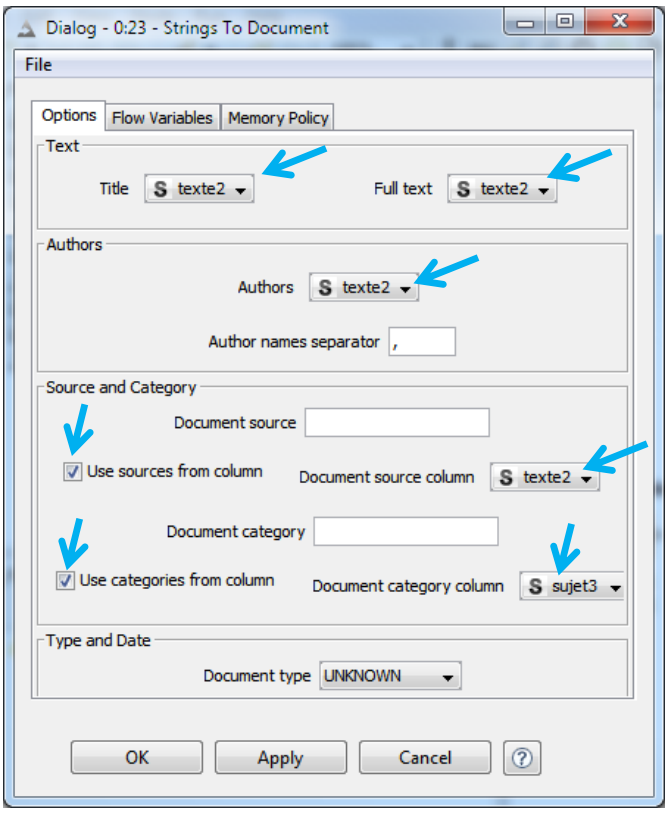
Dans le cadre de l'apprentissage supervisé, nous souhaitons utiliser les « textes » pour prédire les « sujets ». Il est donc nécessaire de réunir les deux vecteurs dans une table de données unique en veillant à faire correspondre les lignes (ROW ID). Nous utilisons l'outil DATA MANIPULATION / COLUMN / SPLIT & COMBINE / JOINER pour ce faire.



3.5 Préparation et nettoyage des textes

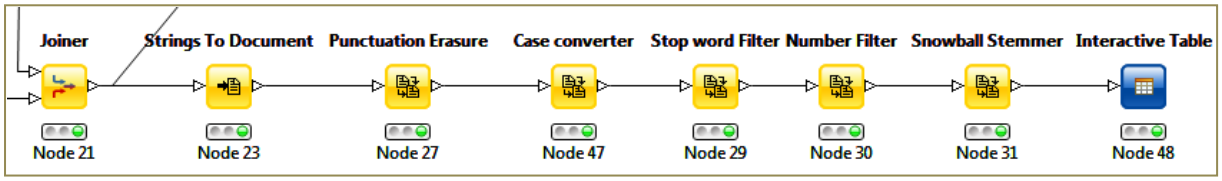
Préalablement à la création de la matrice documents-termes, il convient de nettoyer le texte. Cela passe par plusieurs étapes :

- Convertir le texte au format interne « document » en précisant les différentes parties. Nous utilisons le composant KNIME LABS / TEXT PROCESSING / TRANSFORMATION / STRINGS TO DOCUMENT. Voici le paramétrage utilisé. La désignation de SUJET3 comme indicateur de catégorie des documents est essentielle pour la suite.



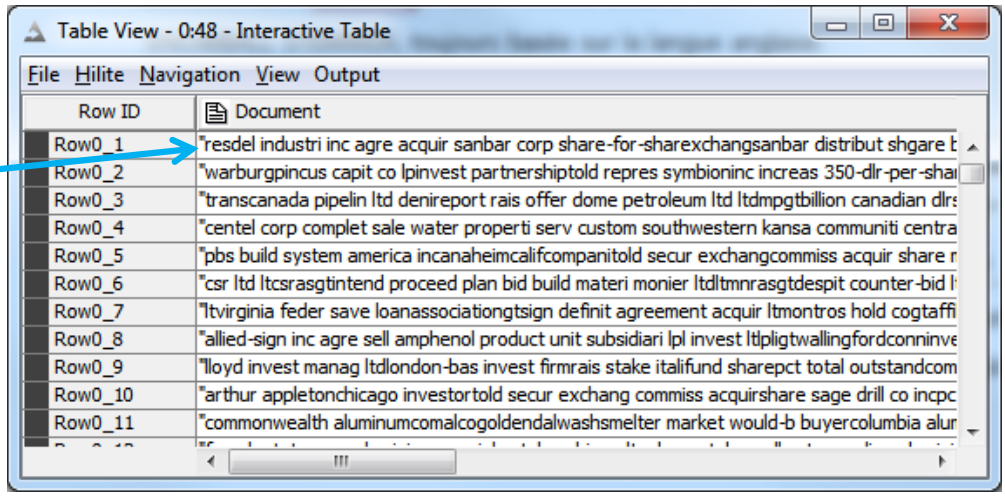
- Supprimer la ponctuation du document à l'aide du composant KNIME LABS / TEXT PROCESSING / PREPROCESSING / PUNCTUATION ERASURE.
- Convertir le texte en minuscule à l'aide de DATA MANIPULATION / COLUMN / TRANSFORM / CASE CONVERTER.
- Supprimer les mots vides (« stop words ») avec KNIME LABS / TEXT PROCESSING / PREPROCESSING / STOP WORD FILTER en utilisant la liste interne des mots en anglais (Use build in list – Stopword lists : English).
- Retirer les nombres avec KNIME LABS / TEXT PROCESSING / PREPROCESSING / NUMBER FILTER.
- Effectuer le « stemming » avec KNIME LABS / TEXT PROCESSING / PREPROCESSING / SNOWBALL STEMMER, basée sur la langue anglaise.

Voici le diagramme de traitements à partir de JOINER.



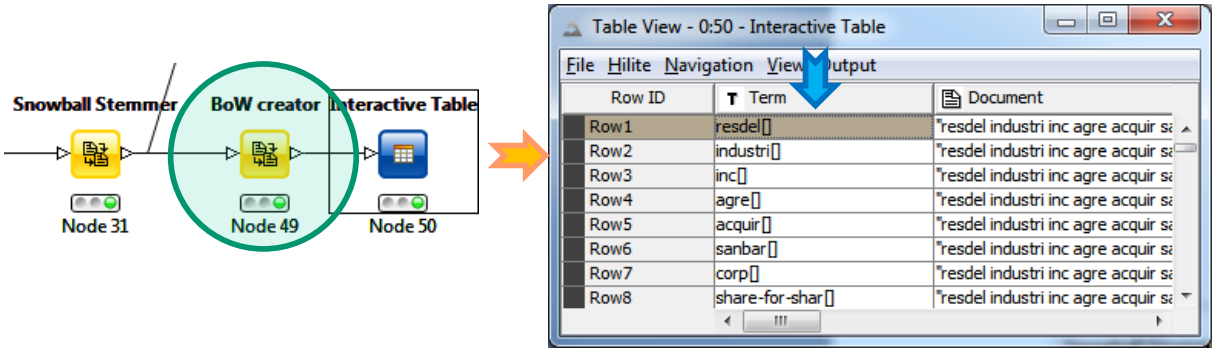
Le nettoyage est particulièrement drastique. On reconnaît à peine le texte si l'on effectue le parallèle pour la 1ère nouvelle par exemple. Mais il s'avère nécessaire pour distinguer l'information utile du « bruit » inhérent à tout objet de communication. A priori, l'information nécessaire à l'analyse statistique y figure. Il faut l'espérer en tous les cas sinon tout traitement ultérieur serait vain.

Resdel Industries Inc said it has agreed to acquire San/Bar Corp in a share-for-share exchange, after San/Bar distributes all shgares of its Break-Free Corp subsidiary to San/Bar shareholders on a share-for-share basis.
The company said also before the merger, San/Bar would Barry K. Hallamore and Lloyd G. Hallamore, San/Bar's director of corporate development, 1,312,500 dlrs and 1,087,500 dlrs respectviely under agreements entered into in October 1983.



3.6 Extraction des termes

Nous utilisons l'outil KNIME LABS / TEXT PROCESSING / TRANSFORMATION / BOW CREATOR pour extraire la liste des termes (mots) apparaissant au moins une fois dans l'ensemble des documents.



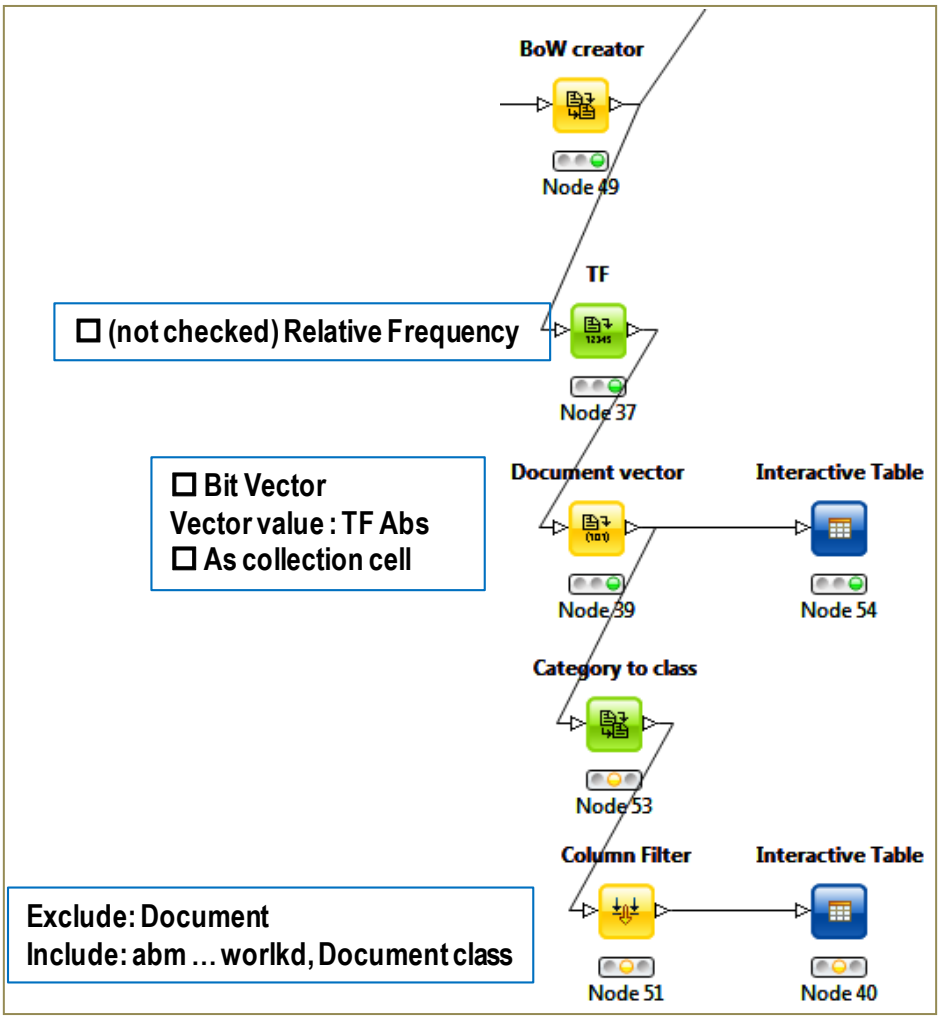
Nous obtenons un tableau avec la liste des termes et les documents où ils apparaissent.

3.7 Choix de la pondération TF et création de la matrice DT

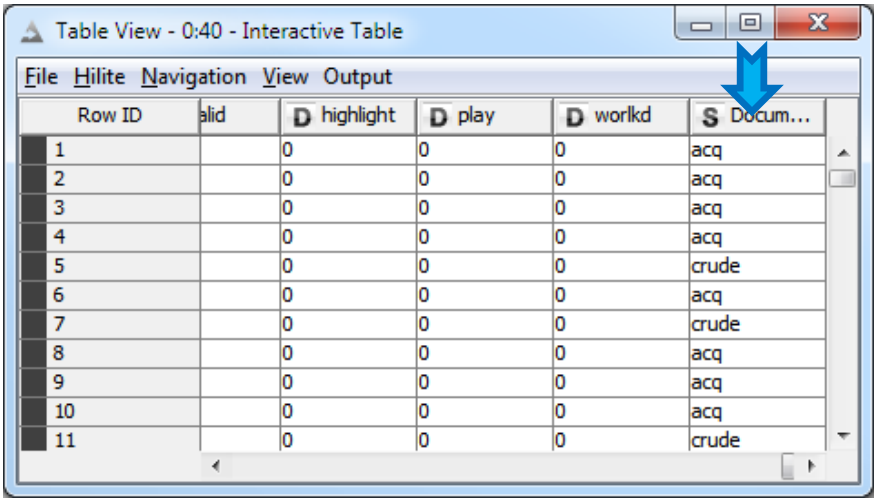
Nous choisissons la pondération TF (Term Frequency) dans un premier temps. Il s'agit simplement de compter le nombre d'apparition des termes dans chaque document. Nous avons besoin des opérateurs suivants pour créer la matrice DT (documents-termes) :

- KNIME LABS / TEXT PROCESSING / FREQUENCIES / TF s'occupe du comptage des termes, nous prenons la fréquence absolue.
- KNIME LABS / TEXT PROCESSING / TRANSFORMATION / DOCUMENT VECTOR se charge de créer la matrice DT, nous spécifions en VECTOR VALUE la colonne TF ABS générée à l'étape précédente.
- KNIME LABS / TEXT PROCESSING / MISC / CATEGORY TO CLASS permet de transformer la catégorie assignée à chaque document (voir le composant STRINGS TO DOCUMENT introduit dans la section 3.5) en attribut classe c.-à-d. la variable cible qui sera utilisée pour l'apprentissage supervisé.
- Enfin, DATA MANIPULATION / COLUMN / FILTER / COLUMN FILTER permet d'exclure la colonne « document » représentant le texte brut de l'ensemble de données.

Voici la séquence des traitements. Nous mettons en avant les paramètres modifiés par rapport à la configuration par défaut de chaque composant.



Avec INTERACTIVE TABLE, nous visualisons la matrice documents-termes avec, en dernière position, la classe d'appartenance de chaque document.

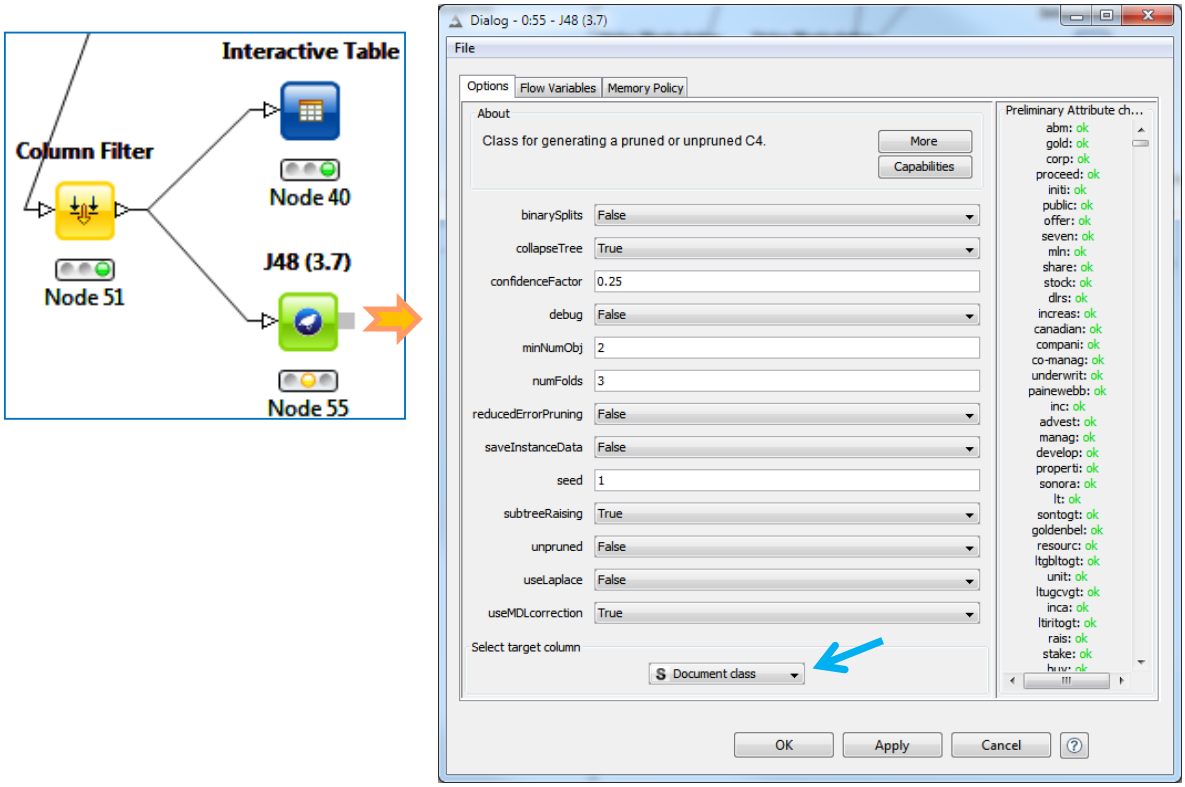


Row ID	alid	highlight	play	worlkd	Docum...
1		0	0	0	acq
2		0	0	0	acq
3		0	0	0	acq
4		0	0	0	acq
5		0	0	0	crude
6		0	0	0	acq
7		0	0	0	crude
8		0	0	0	acq
9		0	0	0	acq
10		0	0	0	acq
11		0	0	0	crude

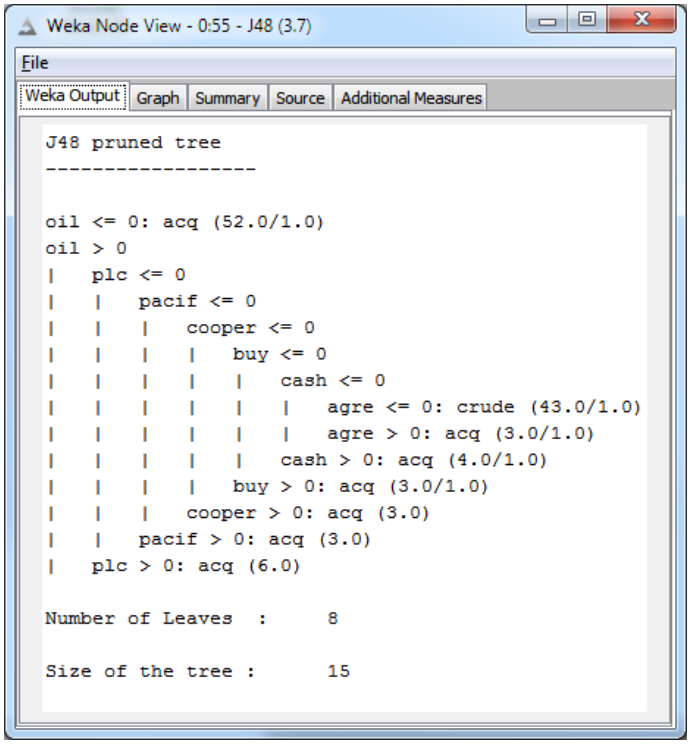
Nous disposons de n = 117 observations (bien évidemment), et de 2418 termes/descripteurs (+ l'attribut classe) pour l'apprentissage.

3.8 Construction du classifieur – Arbre de décision

Nous décidons d'utiliser la méthode J48 de l'extension WEKA qu'il faut installer au préalable. La variable cible « Document Class » est spécifiée lors du paramétrage de l'outil.



A l'exécution, nous obtenons l'arbre suivant.




```
J48 pruned tree
------------------

oil <= 0: acq (52.0/1.0)
oil > 0
|   plc <= 0
|   |   pacif <= 0
|   |   |   cooper <= 0
|   |   |   |   buy <= 0
|   |   |   |   |   cash <= 0
|   |   |   |   |   |   agre <= 0: crude (43.0/1.0)
|   |   |   |   |   |   agre > 0: acq (3.0/1.0)
|   |   |   |   |   cash > 0: acq (4.0/1.0)
|   |   |   |   buy > 0: acq (3.0/1.0)
|   |   |   cooper > 0: acq (3.0)
|   |   pacif > 0: acq (3.0)
|   plc > 0: acq (6.0)

Number of Leaves  :     8

Size of the tree :      15
```

Nous constatons que :

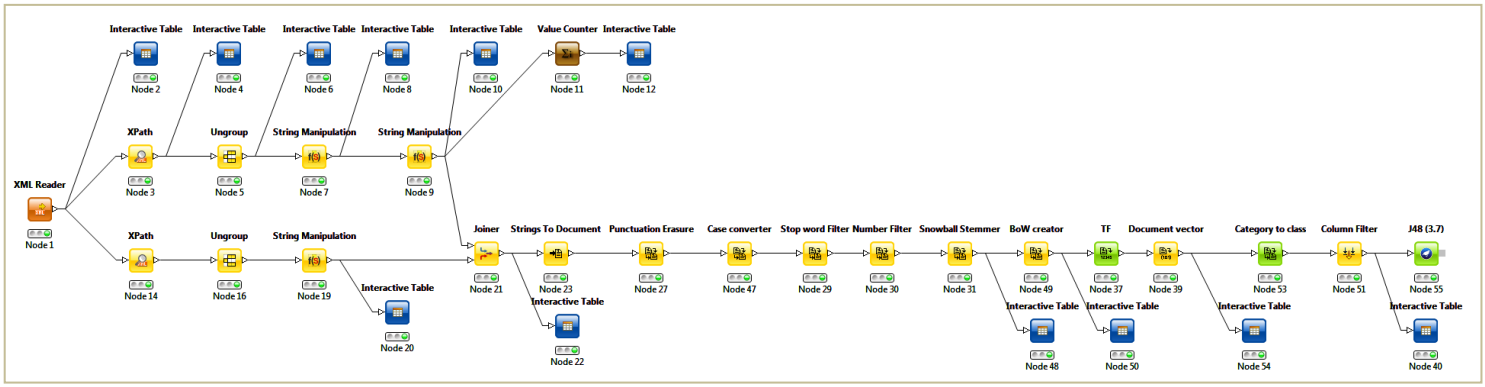
(1) Le document est associé au sujet « **crude** » : **SI** le terme « oil » apparaît (au moins une fois) **ET** (plc, pacif, cooper, buy, cash, agre) n'apparaissent pas (de manière simultanée) dans le document.

(2) Dans tous les autres cas, le document est affecté à « **acq** ».

Ces règles d'affectation peuvent être facilement déployées dans un système d'information.

3.9 Premier bilan

Récapitulons les traitements réalisés pour obtenir ce résultat. Le diagramme de traitements est relativement conséquent. Les composants INTERACTIVE TABLE servent avant tout à surveiller le bon déroulement des opérations à chaque étape[10].



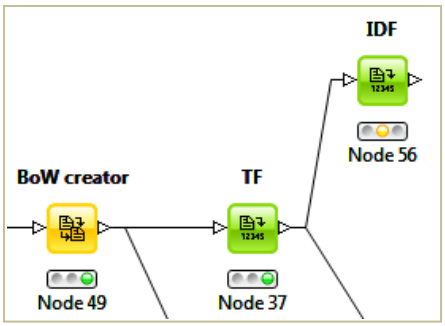
Comme toujours, l'enchaînement paraît très simple après coup. Le plus difficile finalement sous KNIME aura été d'identifier le bon outil pour chaque traitement et de spécifier le paramétrage adéquat. La nécessité de catégoriser au préalable les documents (section 3.5) afin de pouvoir générer la variable cible en fin de processus n'a pas été simple à trouver. Je ne sais pas s'il existe une autre manipulation, plus simple, plus facile à expliquer surtout.

3.10 Variante – Utilisation de la pondération TF-IDF

Maintenant que l'on sait faire, on peut essayer d'aller plus loin. On se propose d'introduire la pondération TF-IDF. Voyons comment y parvenir sous KNIME.

Calcul de l'IDF (Inverse Document Frequency) des termes. Nous utilisons l'outil KNIME LABS / TEXT PROCESSING / FREQUENCIES / IDF pour calculer l'inverse de la fréquence des termes dans les documents. Nous le plaçons à la suite de TF puisque nous allons les combiner par la suite. Il n'y a pas de paramétrage particulier à préciser.

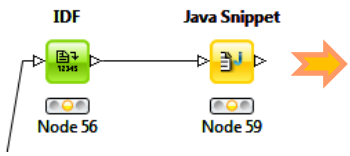
[10] Le diagramme au format KNIME est intégré au fichier archive qui accompagne ce tutoriel.

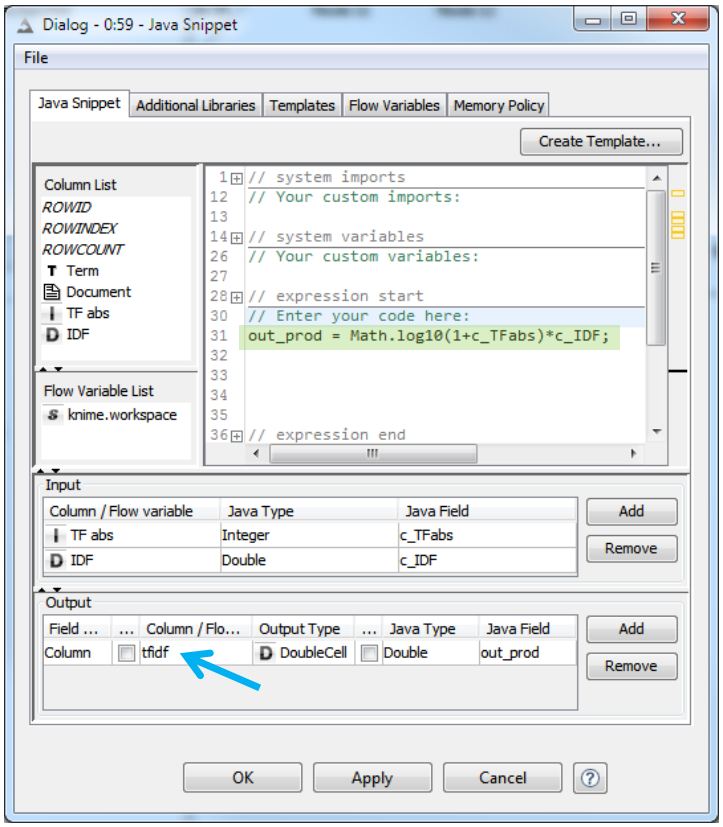


Construction de la pondération TF-IDF. Il ne reste plus qu'à associer les 2 indicateurs TF et IDF. Nous utilisons la formule suivante dans ce tutoriel[11] :

$$TFIDF = LOG_{10}(1+ TF) * IDF$$

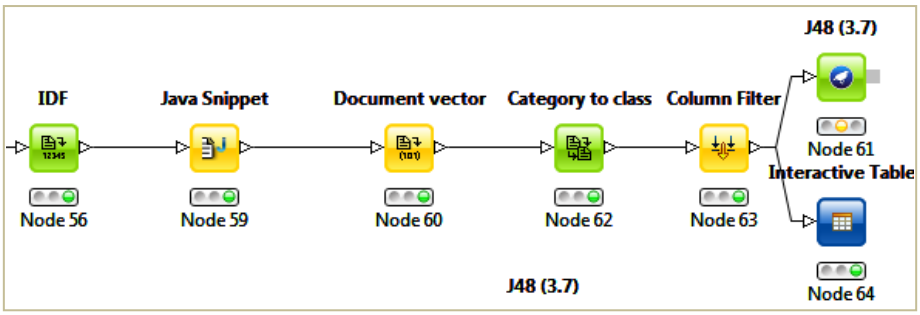
Nous utilisons le composant MISC / JAVA SNIPPET / JAVA SNIPPET pour construire la nouvelle variable. Il semble que cet outil soit particulièrement puissant et permette une très grande variété d'opérations et autres manipulations. En effet, il est capable de prendre en compte du code JAVA c.-à-d. nous pouvons programmer des actions complexes en langage JAVA. Manifestement, nous n'utilisons qu'une infime partie de ses possibilités ici. Voici le paramétrage dans notre cas.





[11] D'autres formules sont possibles, voir http://en.wikipedia.org/wiki/Tf-idf

Suite des traitements. Enfin, comme précédemment, nous réalisons la suite des traitements en plaçant les outils DOCUMENT VECTOR (**Vector value = TFIDF** ; attention au paramétrage), CATEGORY TO CLASS, COLUMN FILTER et J48 pour produire le modèle prédictif.



Voici une vue partielle du tableau de données utilisé, les pondérations sont bien évidemment différentes des précédentes (TF), elles sont définies dans $\mathbb{R}^+$ maintenant.

Table View - 0:64 - Interactive Table

File Hilite Navigation View Output

Row ID	abm	gold	corp	proceed	initi	public	offer	seven	mln
118	1.751	1.213	0.495	0.873	0.706	0.596	0.66	0.764	0.304
119	0	0	0	0	0	0	0.451	0	0.415
120	0	0	0.41	0.596	0	0.596	0	0	0.535
121	0	0	0	0	0	1.055	0.798	0	0.608
122	0	0	0.28	0	0	0	0	0	0.415
123	0	0	0.28	0	0	0	0.451	0	0
124	0	0	0	0	0	0	0	0	0.304
125	0	0	0	0	0	0	0	0	0
126	0	0.872	0.559	0	0	0	0	0	0.304
127	0	0	0	0	0	0	0	0	0.484
128	0	0	0	0	0	0	0	0	0
129	0	0	0.28	0	0	0	0	0	0.304

L'arbre est exactement le même que précédemment. Ce n'est pas étonnant puisque les règles étaient exclusivement basées sur la présence ou l'absence des termes (TF > 0 ou pas).

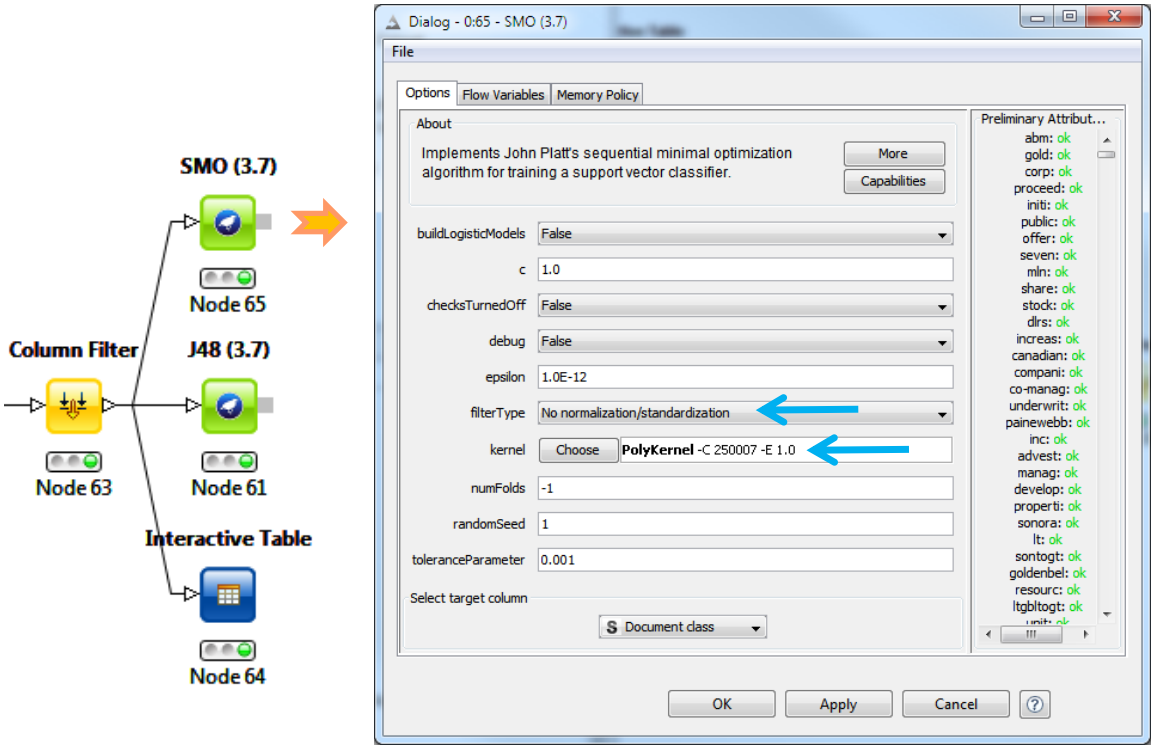
Weka Node View - 0:61 - J48 (3.7)

File

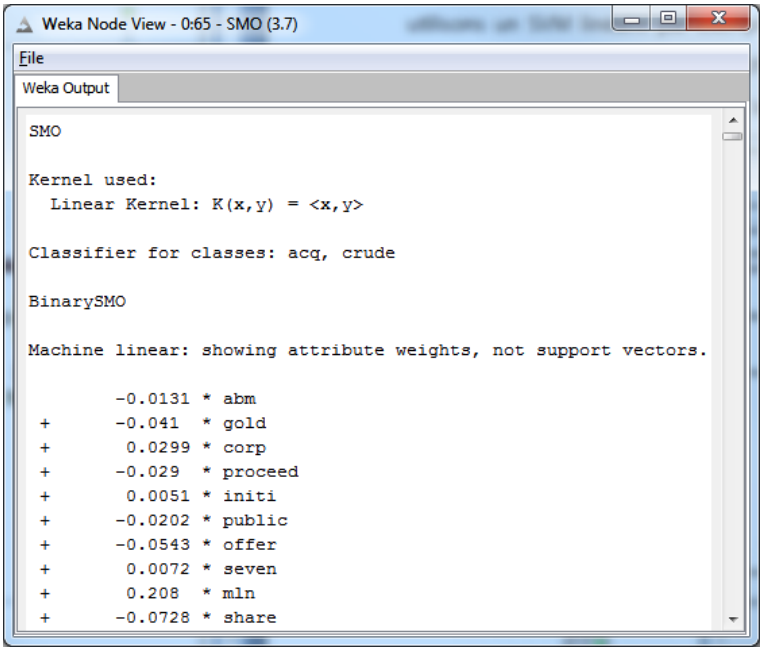
Weka Output | Graph | Summary | Source | Additional Measures

```
J48 pruned tree
------------------

oil <= 0: acq (52.0/1.0)
oil > 0
|   plc <= 0
|   |   pacif <= 0
|   |   |   cooper <= 0
|   |   |   |   buy <= 0
|   |   |   |   |   cash <= 0
|   |   |   |   |   |   agre <= 0: crude (43.0/1.0)
|   |   |   |   |   |   agre > 0: acq (3.0/1.0)
|   |   |   |   |   cash > 0: acq (4.0/1.0)
|   |   |   |   buy > 0: acq (3.0/1.0)
|   |   |   cooper > 0: acq (3.0)
|   |   pacif > 0: acq (3.0)
|   plc > 0: acq (6.0)

Number of Leaves  :     8

Size of the tree :      15
```

Puisque nous disposons d'une pondération plus élaborée. Voyons ce qu'il en est lorsque nous utilisons un SVM (support vector machine) linéaire par exemple. Nous introduisons le composant WEKA / WEKA (3.7) / CLASSIFICATION ALGORITHMS / FUNCTIONS / SMO (3.7). Nous spécifions un noyau linéaire et choisissons de ne pas normaliser les données.



Voici les premiers coefficients de la fonction de classement.



Voici le « workflow » retraçant l'analyse dans sa globalité. Il est assez impressionnant !

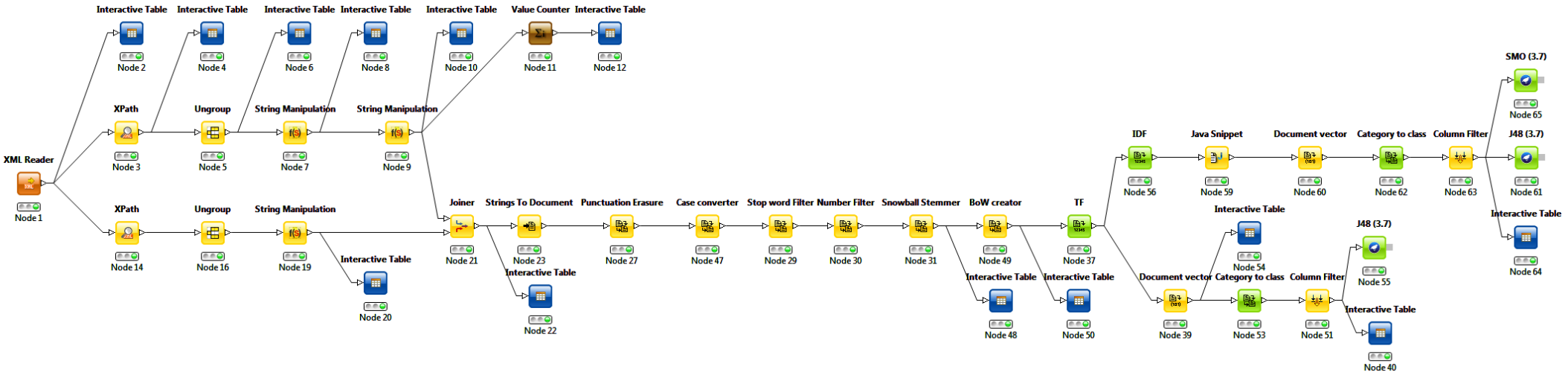


Figure 1 - Analyse complète menée sous KNIME

3.11 Conclusion

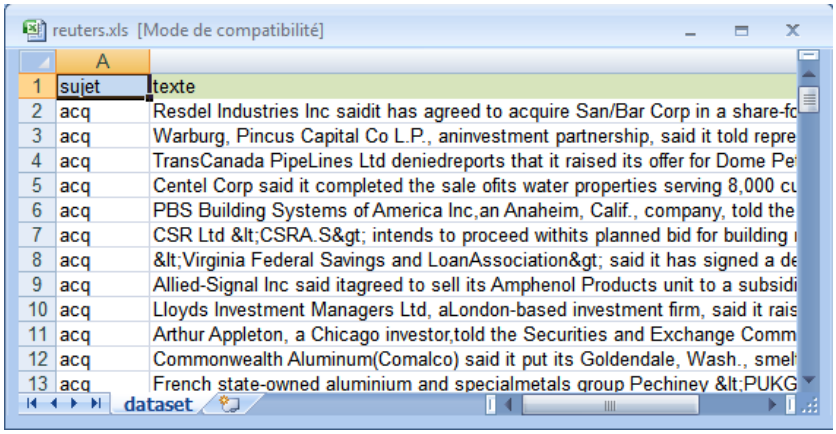
J'ai fait au plus simple dans ce tutoriel. Le classifieur est élaboré à partir de l'ensemble des documents disponibles. Si l'on souhaite aller plus loin et mesurer les performances en prédiction, il faudrait mettre en place une procédure d'évaluation basée sur des techniques de ré-échantillonnage comme la validation croisée[12]. KNIME est tout à fait équipé pour cela[13]. Attention cependant, la procédure doit intégrer le processus d'extraction des termes c.-à-d inclure la chaîne de traitements partant du composant BoW CREATOR jusqu'à la génération du classifieur (J48 ou SMO).

4 Catégorisation de textes avec RapidMiner

RapidMiner est un logiciel reconnu dans la communauté du data mining[14]. La version STARTER est accessible librement sur le site de l'éditeur[15]. Réaliser notre analyse a été vraiment très facile sous RapidMiner… une fois qu'on a compris la logique. Il m'a fallu un peu de temps et beaucoup de recherche sur le web[16] pour comprendre le rôle primordial du composant PROCESS DOCUMENTS FROM DATA dans l'analyse que je souhaitais mener.

4.1 Importation des données

RapidMiner sait lire les fichiers XML avec l'outil « Read XML ». J'ai pourtant préféré utiliser un fichier au format Excel dans cette section. L'idée est de montrer qu'il est finalement possible de stocker des informations textuelles dans tout type de fichier pourvu que l'on soit en mesure de distinguer la catégorie (sujet) du texte brut (texte).



[12] Le schéma apprentissage-test n'est pas réaliste au vu du très faible nombre d'observations disponibles (n = 117).

[13] http://tutoriels-data-mining.blogspot.fr/2008/11/validation-croise-comparaison-de.html

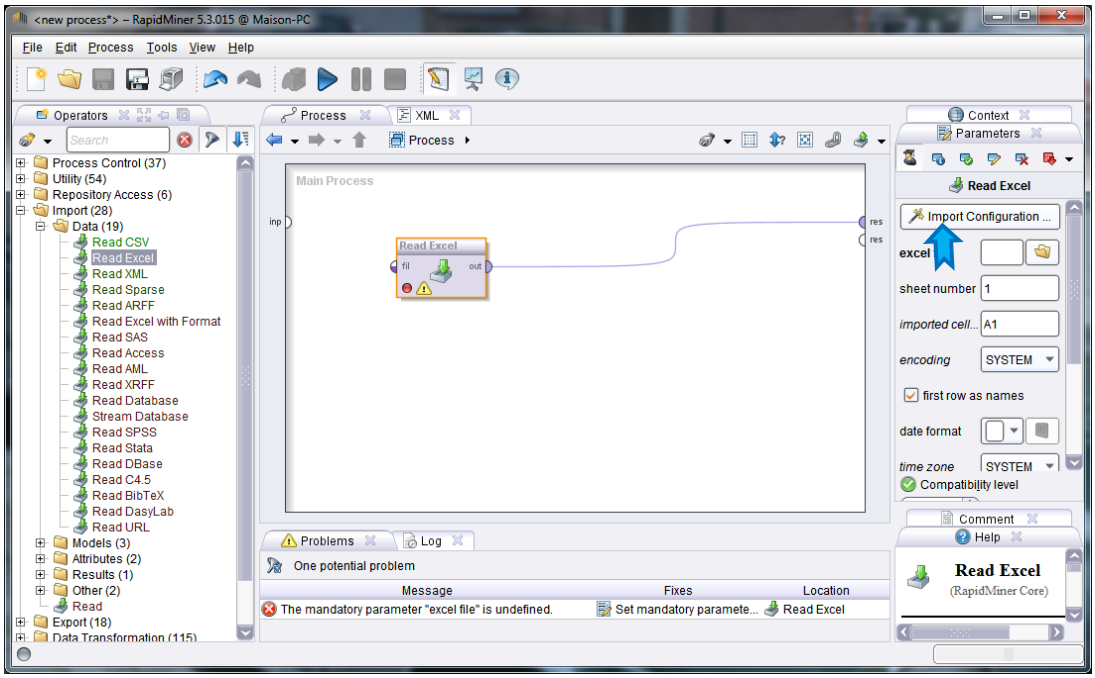
[14] http://www.kdnuggets.com/polls/2013/analytics-big-data-mining-data-science-software.html

[15] http://rapidminer.com/ ; à partir de RapidMiner Studio 6, la version STARTER est bridée malheureusement.

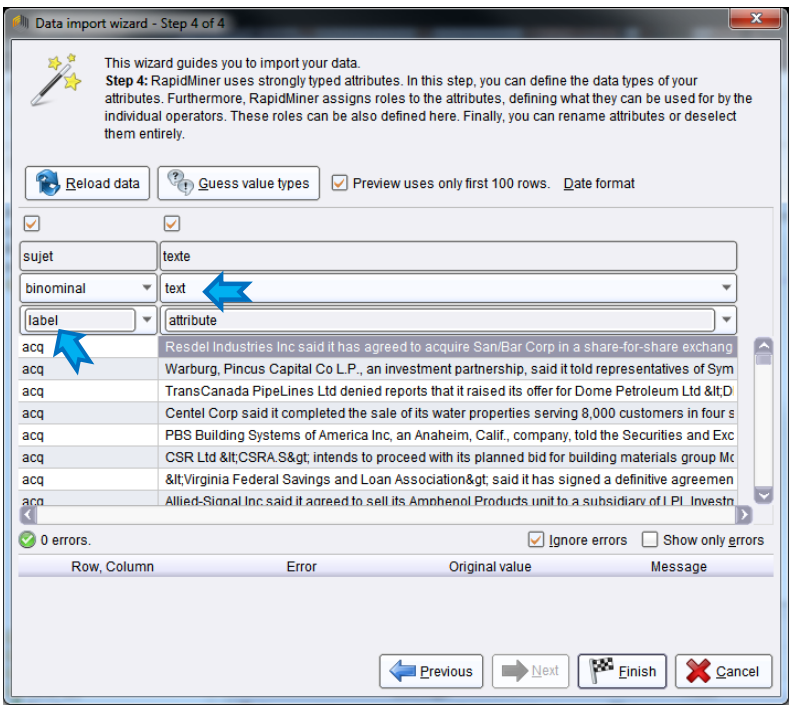
[16] Ce site m'a énormément aidé : http://vancouverdata.blogspot.fr/2010/11/text-analytics-with-rapidminer-loading.html

L'attribut cible est dans la première colonne du fichier « reuter.xls », le texte dans la seconde.

Après avoir démarré RapidMiner, nous créons un nouveau « Process » en cliquant sur le menu FILE / NEW PROCESS. L'interface globale et le mode de fonctionnement de RapidMiner sont identiques à ceux de Knime. Nous utilisons le composant « Read Excel » pour accéder aux données, nous cliquons sur le bouton « Import Configuration Wizard ».

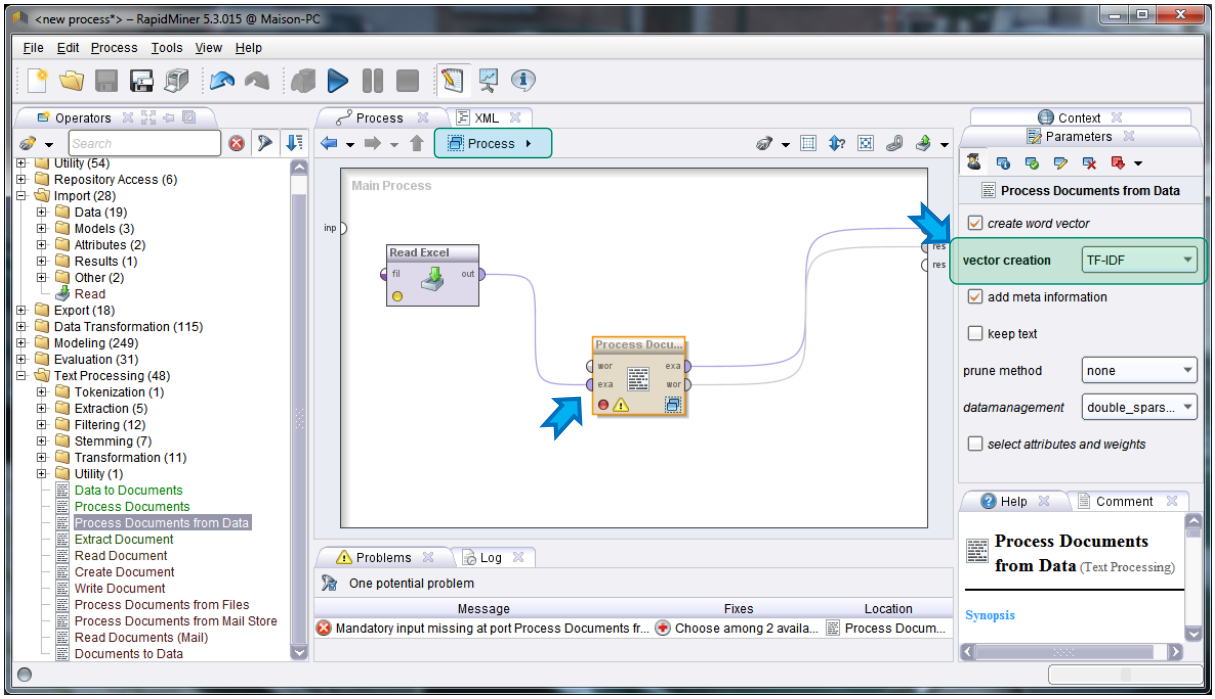


Une étape clé du processus d'importation permet de préciser le rôle des colonnes : « sujet » correspond à l'étiquette des documents, « texte » est de type « text ».



4.2 Le composant « Process Documents from Data »

Nous introduisons ensuite le composant TEXT PROCESSING / PROCESS DOCUMENTS FROM DATA. Nous lui connectons « Read Excel » sur l'entrée « example set ». La pondération TF-IDF est sélectionnée par défaut.

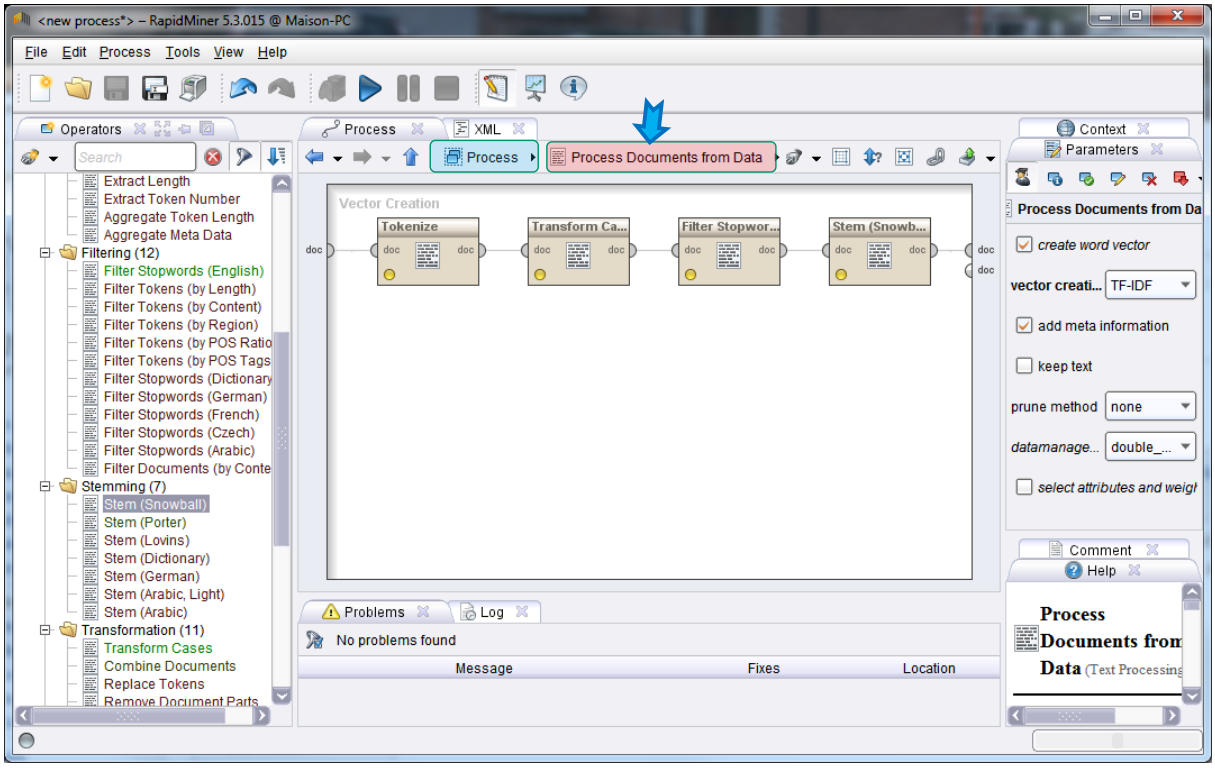


PROCESS DOCUMENTS FROM DATA est en réalité un outil composite qui se charge de générer directement la matrice documents-termes, en lui adjoignant la colonne « sujet » puisque nous avions pris soin de la typer comme « label » durant l'importation ci-dessus.

On accède à la structure interne du composant en double-cliquant dessus. Nous pouvons alors préciser la succession de traitements que nous souhaitons effectuer durant la génération de la matrice DT. Pour nous, il s'agit surtout d'introduire les opérateurs de nettoyage de texte : TEXT PROCESSING / TOKENIZATION / TOKENIZE pour identifier les mots dans le texte[17] ; TEXT PROCESSING TRANSFORMATION / TRANSFORM CASES pour modifier la casse du texte (tout mettre en minuscule) ; TEXT PROCESSING / FILTERING / FILTER STOPWORDS (ENGLISH) pour éliminer les mots vides ; TEXT PROCESSING / STEMMING / STEM (SNOWBALL) pour ne retenir que la racine des mots dans la génération des termes.

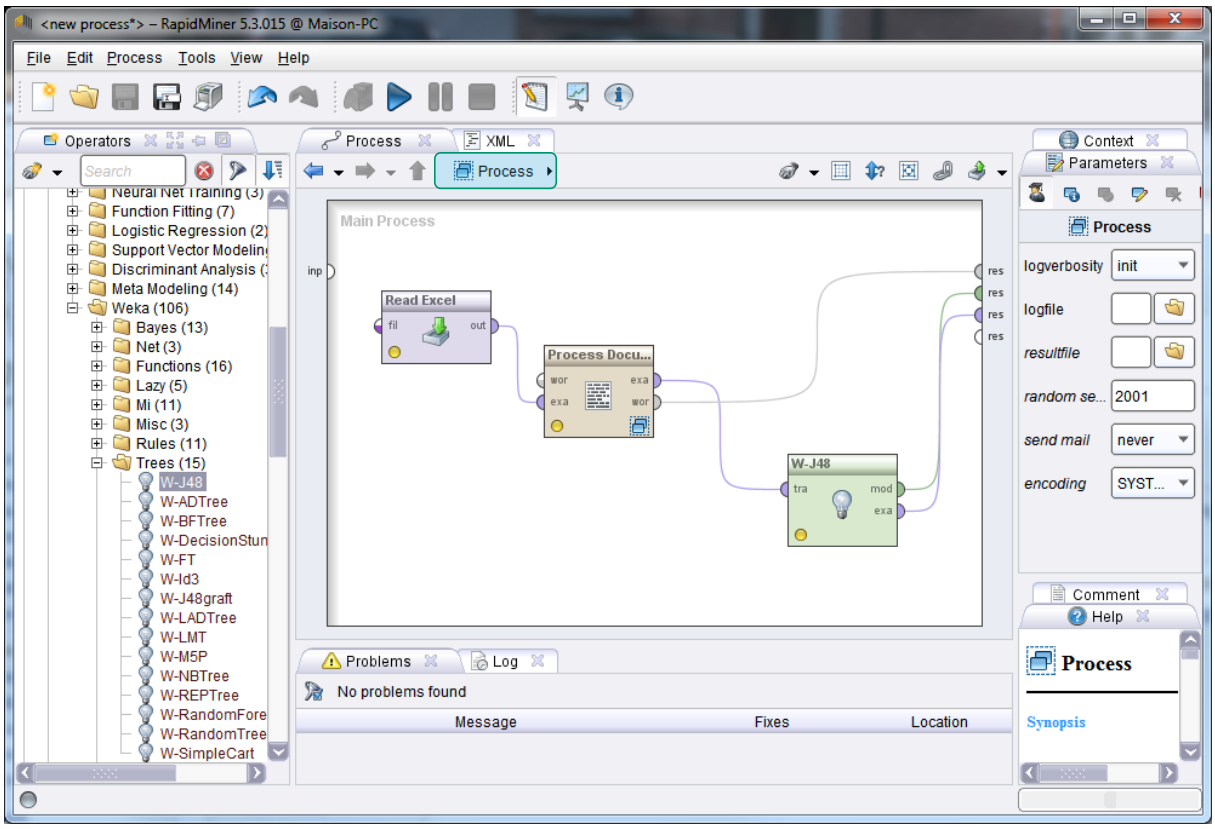
Voici le diagramme. Notez bien que nous visualisons **l'intérieur** du composant PROCESS DOCUMENTS FROM DATA dans la copie d'écran ci-dessous.

[17] http://en.wikipedia.org/wiki/Tokenization



4.3 Choix du classifieur – Weka J48

Nous cliquons sur la flèche « ↑ » pour revenir au niveau principal. Nous insérons la méthode W-J48 importée de l'extension WEKA pour générer l'arbre de décision.

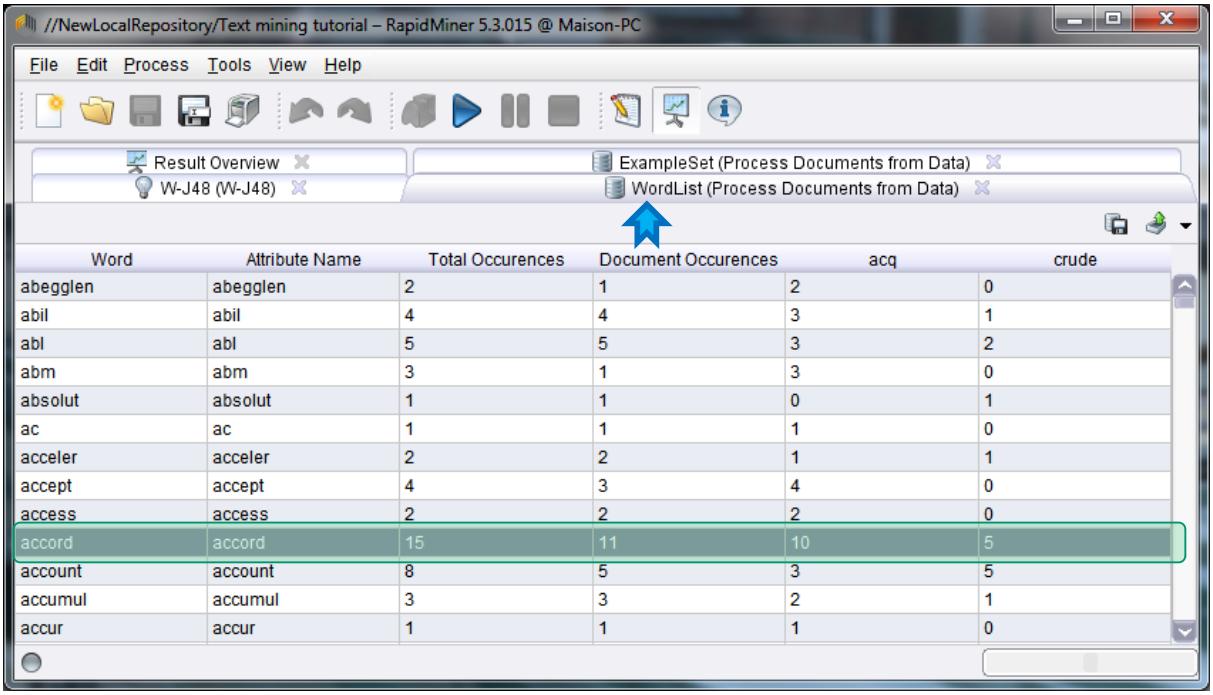


Remarquons les différentes connexions, notamment celles qui sont reliées à la sortie du diagramme. Elles définissent les résultats consultables à l'issue des calculs.

4.4 Résultats

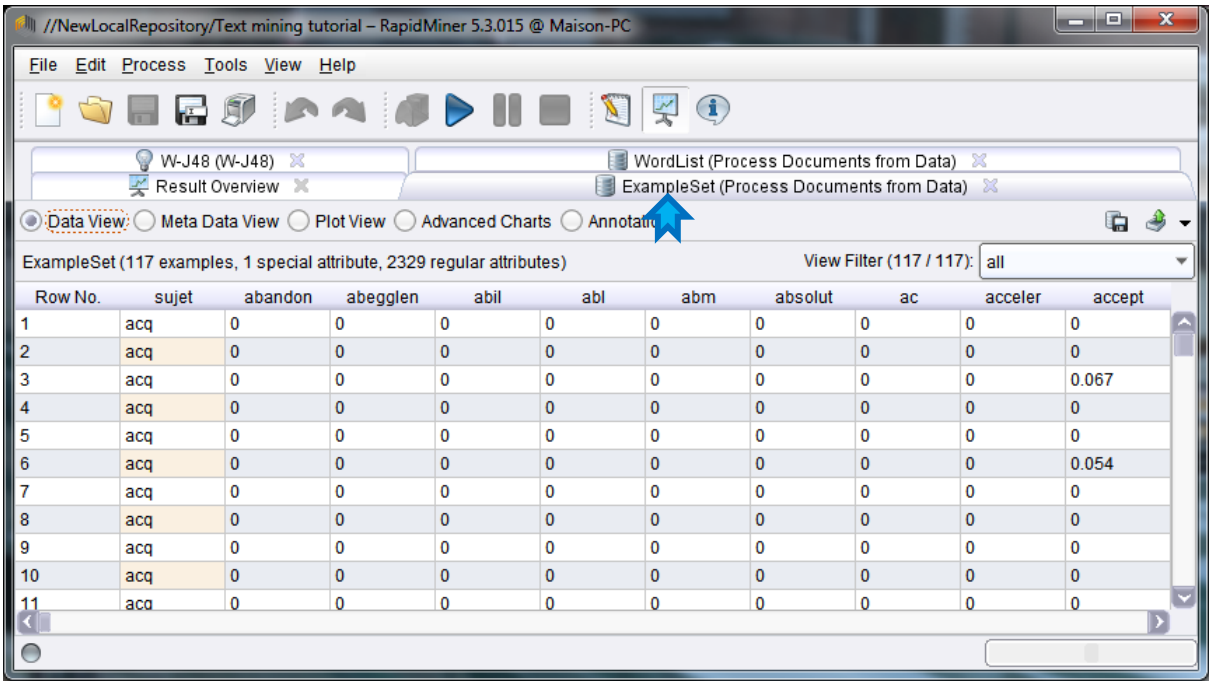
Il ne nous reste plus qu'à exécuter le « process ». Une boîte de dialogue nous invite à effectuer une sauvegarde. Nous acquiesçons en spécifiant le nom de projet « Text mining tutorial ». Plusieurs résultats répartis dans des onglets retiennent notre attention.

Wordlist. Les termes sont énumérés dans cet onglet. Par exemple, le terme « accord » est présent dans 11 documents, il apparaît 15 fois en tout (il peut apparaître plusieurs fois dans un document), 10 sont associées au sujet « acq », 5 à « crude ». Ces informations sont loin d'êtres anodines. Elles nous donnent des indications sur la pertinence du terme dans la désignation des classes.

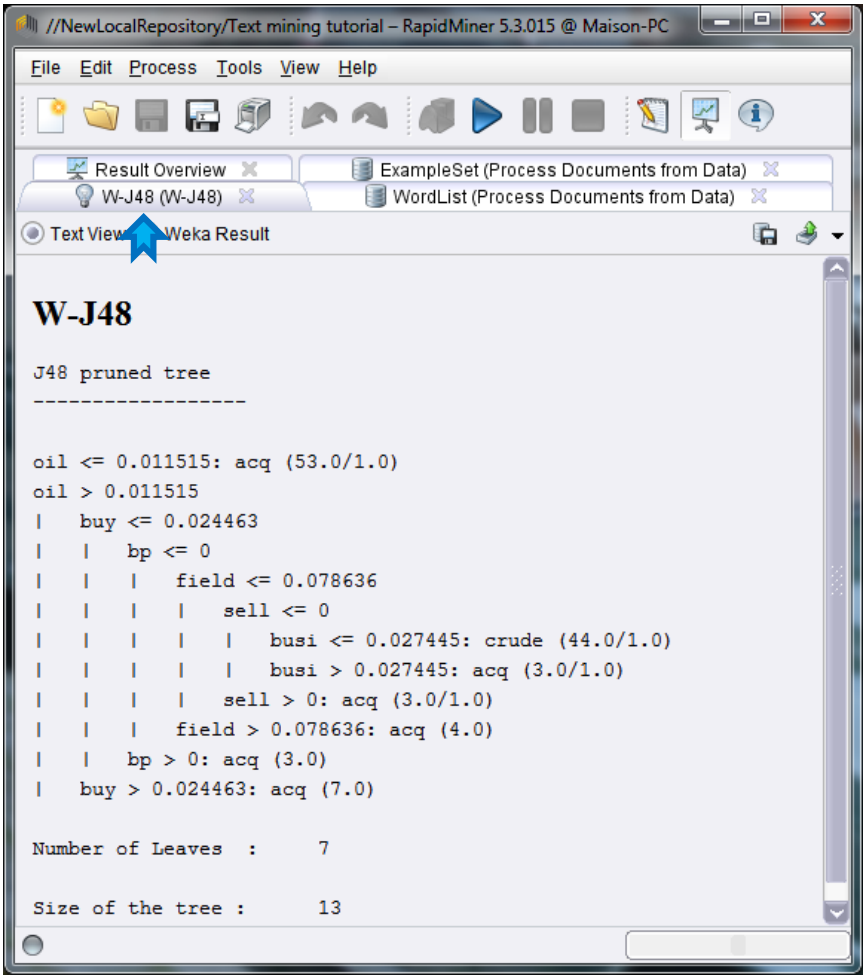


Word	Attribute Name	Total Occurences	Document Occurences	acq	crude
abegglen	abegglen	2	1	2	0
abil	abil	4	4	3	1
abl	abl	5	5	3	2
abm	abm	3	1	3	0
absolut	absolut	1	1	0	1
ac	ac	1	1	1	0
acceler	acceler	2	2	1	1
accept	accept	4	3	4	0
access	access	2	2	2	0
accord	accord	15	11	10	5
account	account	8	5	3	5
accumul	accumul	3	3	2	1
accur	accur	1	1	1	0

ExampleSet. Nous y visualisons la matrice documents-termes, 2329 termes ont été générés. Les résultats sont légèrement différents de ceux de Knime : d'une part, parce que nous n'avons pas introduit les mêmes opérateurs de nettoyage de texte (ex. « remove punctuations » sous Knime, etc.) ; d'autre part, parce que les algorithmes de nettoyage ne sont vraisemblablement pas implémentés de manière strictement identique (ex. stemming).



Arbre de décision. Enfin, nous disposons de l'arbre de décision J48.



Etrangement, l'arbre est différent de celui de Knime, pourtant les deux logiciels reposent sur le même algorithme de génération d'arbre (J48 de WEKA). Cela s'explique par les différences entre les matrices de documents-termes, qui ne s'appuient pas une liste identique de termes extraits : 2418 termes pour Knime, 2329 pour RapidMiner. Une étude complémentaire intéressante serait de comparer ces 2 listes.

5 Conclusion

« Qu'importe le flacon pourvu qu'on ait l'ivresse » a-t-on coutume de dire. Ce dicton s'applique très bien ici. Le fond de l'affaire est le même, on cherche à produire la matrice documents-termes à partir des données textuelles afin de pouvoir appliquer les techniques de data mining. Mais les deux logiciels organisent différemment les traitements. Sous Knime, l'approche est très didactique, les actions sont décomposées à l'extrême, au risque de perdre l'utilisateur par la profusion des outils à placer dans l'espace de travail. Sous RapidMiner, on dispose d'un composant central « qui fait tout ». On est perdu si on ne le sait pas. Il fallait savoir également qu'il est possible de le paramétrer en y insérant des opérations intermédiaires de nettoyage de texte. A la sortie, tout comme R avec le package 'tm' (c'est l'objet du travail sur machine pour le cours de Text Mining), nous pouvons produire un classifieur permettant d'affecter automatiquement les sujets aux textes. C'est ce qui importe finalement.